DESARROLLO DE PLUGIN PARA EL ANÁLISIS DE DATOS GENERADOS POR SECUENCIACIÓN DE ALTO RENDIMIENTO

JUAN CAMILO QUINTERO LOPEZ

Director:

PhD. CÉSAR JESÚS PARDO CALVACHE

Codirector:

PhD. JORGE DUITAMA CASTELLANOS



UNIVERSIDAD DE SAN BUENAVENTURA

FACULTAD DE INGENIERÍA

PROGRAMA INGENIERÍA DE SISTEMAS

SANTIAGO DE CALI

2013

DESARROLLO DE PLUGIN PARA EL ANÁLISIS DE DATOS GENERADOS POR SECUENCIACIÓN DE ALTO RENDIMIENTO

JUAN CAMILO QUINTERO LOPEZ

TRABAJO DE TESIS PARA OPTAR EL TÍTULO DE INGENIERO DE SISTEMAS

Director:

PhD. CÉSAR JESÚS PARDO CALVACHE

Codirector:

PhD. JORGE DUITAMA CASTELLANOS



UNIVERSIDAD DE SAN BUENAVENTURA

FACULTAD DE INGENIERÍA

PROGRAMA INGENIERÍA DE SISTEMAS

SANTIAGO DE CALI

2013

Nota de Aceptación:

Aprobado por el Comité de Trabajos de Grado en cumplimiento de los requisitos exigidos por la Universidad de San Buenaventura Seccional Cali para optar al título de Ingeniero de Sistemas.

____

PhD. César Jesús Pardo Calvache
Director Proyecto de Grado

____

Jurado 1

____

Jurado 2

Santiago de Cali, 21 de noviembre de 2013

DEDICATORIA

Quiero dedicarle esta tesis principalmente a mi mamá; gracias mamá por tu apoyo incondicional estos 22 años, sin tu ayuda no sería posible ningún de los éxitos que he conseguido hasta ahora, te quiero con todo mi corazón.

AGRADECIMIENTOS

En este capítulo de mi tesis quisiera extender mis más sinceros agradecimientos a Jorge Duitama por su inmensa comprensión y colaboración, gracias Jorge por darme la oportunidad de hacer parte de una empresa maravillosa como lo es el CIAT, por ser más que un jefe un líder del cual he aprendido muchísimas cosas que me servirán tanto para mi vida laboral, como en sociedad. No ha sido un camino fácil desde que llegue a CIAT me enfrentado al reto de aprender conceptos de una carrera que no estudie como biología y sumar mayores conocimientos para mi carrera sistemas, logrando una unificación de ambas carreras en una rama fantástica desde mi concepto como es la bioinformática. En este sentido, debo agradecerte Jorge por permitirme hacer parte de tu proyecto de bioinformática, proyecto con el cual se generó esta tesis.

De igual forma, quiero hacer parte de este agradecimiento a Daniel Cruz, gracias Daniel por tu colaboración a mi formación académica, sin tu ayuda no sería posible la culminación con éxito de esta tesis. También a todos los biólogos con los que he interactuado en CIAT, de alguna manera me han ayudado a resolver dudas con respecto a diferentes temas que se abordan en esta tesis. No puedo dejar de mencionar a CIAT como empresa, gracias por brindarme la oportunidad de trabajar y aprender a diario nuevos conceptos, por un tranquilo ambiente de trabajo, que sin duda hace de CIAT una empresa fantástica.

También, debo agradecer a la universidad San Buenaventura y todos los docentes que aportaron sus conocimientos para lograr una adecuada formación integral. En especial al profesor César Pardo de la Universidad Autónoma de Occidente por su enorme colaboración en la revisión y dirección de esta tesis, gracias totales César.

Por último, quiero darles las gracias a todos y cada uno de mis familiares que me han apoyado en las diferentes etapas de mi vida, en especial a Gloria Lopez y Camilo Montes, este logro también pertenece a todos ustedes.

TABLA DE CONTENIDO

Contenido

INDICE DE TABLAS

INDICE DE ILUSTRACIONES

LISTA DE ANEXOS

GLOSARIO DE TÉRMINOS

1. **Formato SAM**: según lo descrito por Samtools en (Sequence Alignment/Map Format Specication, 2013) SAM acrónimo en inglés de Sequence Alignment / Map format.
   Se trata de un formato de texto delimitado por tabuladores que consiste en una sección de encabezado, que es opcional, y una sección de alineación. Si está presente, el encabezado debe ser antes de la las alineaciones. Las líneas de cabecera empiezan con `@», Mientras que las líneas de alineación no. Cada línea de alineación tiene 11 ELDs obligatorios de información esencial de la alineación como la posición de mapeo y número variable de opciones de información específica a cerca de una secuencia [14].

2. **Formato BAM:** es el mismo formato SAM que se comprime en el formato BGZF [14**¡Error! No se encuentra el origen de la referencia.**].

3. **ADN:** el **ácido desoxirribonucleico**, frecuentemente abreviado como **ADN**, es un ácido nucleico que contiene instrucciones genéticas usadas en el desarrollo y funcionamiento de todos los organismos vivos conocidos y algunos virus, y es responsable de su transmisión hereditaria. El papel principal de la molécula de ADN es el almacenamiento a largo plazo de información [16].

4. **locus genético:** es una localización genética determinada dentro de una secuencia de ADN [1630].

5. **Alelo:** se define como cada una de las formas alternativas de un gen que pueden existir en una localización específica o locus [16].

6. **Genotipo:** combinación de alelos en un locus genético [16].

7. **Genotipo Homocigoto:** un organismo que posee dos copias del mismo alelo, Cuando en un mismo locus genético no hay diferencia de secuencia entre la copia heredada del padre y la copia heredada de la madre, se dice que el sujeto es homocigoto en esa posición [16].

8. **Genotipo Heterocigoto:** si coexisten dos alelos diferentes dentro de un mismo sujeto en ese locus, se dice que el sujeto es heterocigoto [16].

9. **Secuenciación de ADN (sequencing):** Procedimiento analítico que permite determinar la secuencia de aminoácidos de un polipéptido o la secuencia de nucleótidos de una hebra de ADN o de ARN [18].

10. **Oligonucleótido:** es una secuencia corta de ADN o ARN, con cincuenta pares de bases o menos.

11. **Nucleótidos:** son moléculas orgánicas formadas por la unión covalente de un monosacárido de cinco carbonos (pentosa), una base nitrogenada y un grupo fosfato [16].

12. **bases nitrogenadas:** son compuestos orgánicos cíclicos, que incluyen dos o más átomos de nitrógeno, se clasifican en tres grupos, bases púricas o purinas, bases

pirimidinas, la adenina (A) y la guanina (G) son púricas, y la timina (T), la citosina (C) y el uracilo (U) son pirimidínicas. Por comodidad, cada una de las bases se representa por la letra indicada. Las bases A, T, G y C se encuentran en el ADN, mientras que en el ARN en lugar de timina aparece el uracilo [16].

13. **Lectura genómica (*sequence read:*):** lectura de la secuencia (nucleotídica) [24].

14. **Resecuenciación**: La secuenciación de parte del genoma de un individuo con el fin de detectar diferencias de secuencia entre el individuo y el genoma de referencia de las especies [20].

15. **Formato Fasta:** es un formato basado en texto para la representación de cualquiera de las secuencias de nucleótidos o secuencias de péptidos, en el que están representados los nucleótidos o aminoácidos mediante códigos de una sola letra [31].

16. **Formato FASTQ:** se ha convertido en un formato de archivo común para compartir los datos de secuenciación de lectura que combinan la secuencia y se asocia al nivel de calidad de base [31].

17. **Alineamiento de secuencias:** en bioinformática es una forma de representar y comparar dos o más secuencias o cadenas de ADN, ARN, o estructuras primarias proteicas para resaltar sus zonas de similitud, que podrían indicar relaciones funcionales o evolutivas entre los genes o proteínas consultados. Las secuencias alineadas se escriben con las letras (representando aminoácidos o nucleótidos) en filas de una matriz en las que, si es necesario, se insertan espacios para que las zonas con idéntica o similar estructura se alineen [22].

18. **Alineamiento múltiple de secuencias:** es un alineamiento de tres o más secuencias biológicas, generalmente proteínas, ADN o ARN. En general, se asume que el conjunto de secuencias de consulta que se ingresa como entrada (conjunto problema) tienen una relación evolutiva por la cual comparten un linaje y descienden de un ancestro común [23].

19. **Formato vcf:** es un formato de archivo que soporta la llamada de variantes, es flexible y extensible para los datos de variación, tales como (SNP), inserciones / deleciones, las variaciones del número de copias y variantes estructurales [24].

20. **Genoma:** es la totalidad de la información genética que posee un organismo o una especie en particular. El genoma en los seres eucarióticos comprende el ADN contenido en el núcleo, organizado en cromosomas, y el genoma mitocondrial. El término fue acuñado en 1920 por Hans Winkler, profesor de Botánica en la Universidad de Hamburgo, Alemania, como un acrónimo de las palabras gene y cromosoma [16].

21. **NGSTools:** Librería creada por Jorge Duitama, Es un marco integrado para el descubrimiento de variantes genómicas, que utiliza los datos producidos por NGS.

22. **Pipeline:** Flujo de trabajo compuesto por determinados procesos para trabajar con los datos de NGS.

23. **bowtie2:** Es una herramienta ultrarrápida y con memoria-eficiente para la alineación de la secuencia de lecturas con largas secuencias de referencia [39].

24. **SNPs:** Polimorfismo de un nucleótido único, es la forma más sencilla de mutación genética, ya que consisten en el cambio de un soló nucleótido en una secuencia.

25. **CNV:** Una variación del número de copia (CNV) es cuando el número de copias de un gen en particular varía de un individuo a otro.

26. **SANGER:** El método de secuenciación por dideoxinucleótidos, mejor conocido como el método Sanger se basa en el proceso biológico de la replicación del DNA. El método de secuenciación ideado por Sanger está basado en el empleo de dideoxinucleótidos que carecen del grupo hidroxilo del carbono 3', de manera que cuando uno de estos nucleótidos se incorpora a una cadena de DNA en crecimiento, esta cadena no puede continuar elongándose. Esto es así ya que la DNA polimerasa necesita un grupo terminal 3' OH para añadir el siguiente nucleótido y el dideoxinucleótido incorporado carece de este grupo hidroxilo [37].

27. **454:** La secuenciación 454, basada en la secuenciación por síntesis, es posible mediante la plataforma de secuenciación de segunda generación Genome Sequencing FLX. Los nucleótidos fluyen de forma secuencial en un orden fijo a través del soporte de la placa PicoTiter durante una carrera de secuenciación. Durante el flujo de nucleótidos, cientos de miles de perlas unidas a millones de copias de una única molécula de ADN de hebra sencilla son secuenciadas en paralelo. Si un nucleótido es complementario a la cadena molde en algún pocillo, la polimerasa extiende la hebra existente de ADN mediante la adición de nucleótido(s). La adición de uno (o más) nucleótido(s) resulta en una reacción que genera una señal de luz que es recogida por la cámara CCD del equipo. La intensidad de la señal es proporcional al número de nucleótidos incorporados en un solo flujo de nucleótidos [38].

28. **Ilumina:** Es un método de secuenciación que consta de la fragmentación de la muestra de DNA y la unión de los adaptadores, una vez los fragmentos de DNA se unan a los adaptadores se hace una corrida de PCR en puente para determinar la reacción de la secuencia, este proceso se le llama terminadores reversibles, este método utiliza una lámina cubierta por secuencias específicas. Los nucleótidos son detectados uno a uno mediante la luminiscencia que emiten a medida que se van uniendo a la cadena en formación [44].

29. **Estudio de cohorte:** Estudio epidemiológico en el que se hace una comparación de la frecuencia de enfermedad entre dos poblaciones, una de las cuales está expuesta a un determinado factor de exposición (o factor de riesgo), al que no está expuesta la otra [45].

RESUMEN

Esta tesis presenta el desarrollo de la herramienta NGSEP, en este sentido, NGSEP es un plugin de software programado en el lenguaje Java con SWT biblioteca de componentes gráficos. Con esta esta herramienta un usuario final puede realizar diferentes procesos de NGS (Next generation Sequencing). Además ofrece una solución importante para los problemas de integración y usabilidad presente en las actuales herramientas de bioinformática.

CAPÍTULO 1: INTRODUCCIÓN

Desde finales del siglo XVIII el hombre ha indagado sobre las diferencias tanto físicas como de comportamiento de los habitantes del planeta tierra, entre ellos: plantas, animales, bacterias y los propios seres humanos. A partir de la indagación han surgido interrogantes como: ¿En qué radican las diferencias físicas y de comportamiento y por qué nos hacen tan diferentes unos de otros? Este interrogante ha sido abordado por diferentes científicos a lo largo de los años, dejando infinidades de aportes valiosos en el campo de la genética y la biología, como: la identificación de ácidos nucleicos en 1919 y de bases nitrogenadas en 1930, etc. Estos aportes se pueden encontrar por medio de artículos, bases de datos para marcadores moleculares, entre otros.

Ante la necesidad de dar respuestas cada vez más acertadas al interrogante del porqué somos tan diferentes fenotípicamente y genotípicamente unos a otros, es necesario analizar el ADN de cualquier sistema biológico. Bajo esta consigna surgió uno de los primeros y más efectivos métodos de secuenciación del ADN según la comunidad científica del mundo, este método se denominado secuenciación Sanger®, el cual perduró por más de 25 años siendo utilizado en la gran mayoría de laboratorios genéticos alrededor del mundo [9]. Este método permitió secuenciar el *Genoma Humano.* Gracias a la eficacia de Sanger® para secuenciar ADN, en proporciones de quinientos nucleótidos por lectura y con una tasa de error de alrededor del 1%. Además, contribuyó a la comunidad científica en la abstracción de información esencial de sus estudios de caso.

Pese a que Sanger® es eficaz a la hora de secuenciar ADN y genera pocos errores, es demasiado costoso llevar a cabo su implementación. Por este motivo, fue necesario el surgimiento de una nueva era de secuenciación que obtuviera resultados similares o mejores a Sanger® a menor precio de implementación. Esta nueva era se denominó NGS siglas en ingles de Next Generation Sequencing [18].

Desde la introducción de NSG como tecnología se ha visto una gran transformación en la forma como los científicos extraen información genética de los sistemas biológicos, revelando una visión sin límite, acerca del genoma de cualquier especie [10].

Las soluciones alrededor de NGS, han aumentado de manera exponencial. El desarrollo de soluciones ha mejorado la comprensión de la estructura genómica y la función de los distintos organismos con ayuda de los últimos avances en hardware y software, lo que ha permitido realizar resecuenciación sobre información obtenida en el pasado por la tecnología CE-basado en secuenciación de Sanger®.

Con la llegada de NGS como nueva tecnología de secuenciación, se ha generado una enorme cantidad de datos como por ejemplo, lecturas de ADN, genomas secuenciados y archivos con identificación de variantes genómicas. Estos datos son indispensables a la hora de realizar un análisis por parte de los científicos. Bajo esta característica, varias herramientas

bioinformáticas se han desarrollado para llevar a cabo diferentes tipos de análisis. Sin embargo, la mayoría de estas herramientas no son fáciles de instalar, ejecutar, integrar y personalizar sin el apoyo técnico de expertos en bioinformática, lo que produce un cuello de botella para los diferentes esfuerzos de investigación.

En este sentido, el CIAT (Centro internacional de agricultura tropical) ubicado en la zona rural de Palmira, a 17 km de la ciudad de Cali en Colombia. Con sus cultivos (Yuca, Arroz, Frijol y Forrajes) utiliza la tecnología NGS para las investigaciones de mejora de cultivos. Al hacer uso de las herramientas NGS se enfrenta con la problemática de poca usabilidad y de falta de integración entre las herramientas por ejemplo, picardtools y bowtie2, entre otras, esta problemática origina gastos para la organización, porqué debe costear capacitaciones para los científicos en programación, lo que causa perdida de tiempo para los científicos en la elaboración de sus estudios de casos.

Teniendo en cuenta lo anterior, el científico Jorge Duitama Castellanos, doctor en bioinformática e investigador en bioinformática del CIAT, ha desarrollado una librería en java que actualmente se accede por consola denominada NGSTools. NGSTools, se apoya en un flujo de trabajo o pipeline que garantiza la integración de herramientas NGS como bowtie2, garantizando la calidad en los datos producidos en tiempos eficientes [40].

Los resultados producidos por NGSTools contienen la información necesaria de forma actualizada, lo que facilita que dicha información pueda ser posteriormente analizada por los investigadores. Actualmente, NGSTools representa una solución importante para los problemas de integración y tiempos de respuesta pocos eficientes, sin embargo, el hecho que su uso sea por consola no soluciona la problemática de poca usabilidad presente en las herramientas NGS. En ese sentido, surge la necesidad de implementar una interfaz gráfica de usuario (GUI) que integre a NSGTools y garantice aumentar la usabilidad para los usuarios finales.

A grandes rasgos, se considera que los biólogos con problemas en la ejecución de comandos en consola, son usuarios finales y potenciales clientes de la librería NGSTools con implementación de (GUI).

1.1 PROBLEMA Y NECESIDAD

Considerando que la comunidad científica, incluyendo a CIAT, están haciendo uso de la tecnología NGS y sus diferentes tipos de herramientas, se hace indispensable resolver los problemas de poca usabilidad, integración y personalización que se presentan a la hora de implementar el uso de las herramientas NGS como: bowtie2, Picard, NGSTools.

Actualmente la comunidad científica de CIAT accede a las herramientas NGS de manera desordenada y poco eficiente, lo que genera perdida de datos y la imposibilidad de continuar con un flujo de trabajo continuo, ya que las diferentes herramientas no se encuentran integradas. Esto genera resultados faltantes de información valiosa para su posterior análisis.

Esta situación provoca frustración para los científicos, viéndose obligados a capacitarse en cursos de programación que les permitan adquirir los conocimientos suficientes para interactuar con las herramientas NGS. Por otra parte, se aumentan los costos, asociados a la contratación de personal experto y dificulta la realización de los estudios de caso a los científicos en tiempos adecuados.

En este sentido, el Dr. Jorge Duitama Castellanos ha desarrollado y puesto al servicio de CIAT una librería denominada NGSTools, la cual se apoya en un flujo de trabajo o pipeline para garantizar la integración de herramientas que utilizan la tecnología NGS como: Bowtie2, esto con el fin de generar calidad en los datos a producir con un tiempo eficiente.

NGSTools representa una solución importante para los problemas de integración y tiempos de respuesta poco eficientes. Sin embargo, el hecho que su uso sea por consola no soluciona la problemática de poca usabilidad presente en las herramientas NGS.

En consecuencia, se desarrolló un Plug-In el cual presenta un conjunto de interfaces usables para el trabajo con datos de NGS haciendo uso de la librería NGSTools, estas interfaces permiten al usuario monitorizar los procesos una vez estén en ejecución y posteriormente cuando termina la ejecución, como también el manejo de archivos localmente.

1.2 JUSTIFICACIÓN

Teniendo en cuenta la importancia de analizar la variación genética o diversidad genética de un organismo por diferentes motivos que se contextualizan a continuación, es importante resaltar que CIAT con sus cultivos (Yuca, Arroz, Frijol y Forrajes), está en constante búsqueda de mejorar los actuales análisis genéticos con la finalidad de procurar una mayor seguridad alimentaria, reducir el hambre, la pobreza y mejorar la salud humana.

“la diversidad genética es la variedad de alelos y genotipos presentes en una población, especie o grupo de especies, y su importancia radica en que esta es necesaria para que las poblaciones evolucionen y se adapten a las características o cambios en su entorno” [1].

La variación genética se origina principalmente de la mutación y la recombinación que puede dar lugar a mejores (o peores) características adaptativas a las siguientes generaciones [3]. No obstante, la deriva genética, el flujo de genes y la selección que actúa sobre los alelos también pueden introducir nueva variación en las poblaciones y en las especies [5]. Según Suzuki, la variación genética es esencial en la técnica del análisis genético, ya que se puede caracterizar a una especie o una población y se pueden desarrollar marcadores genéticos que sirven para indicar la presencia cercana de un gen de interés en un cromosoma dado.

La diversidad genética juega un papel relevante en el mejoramiento de los cultivos [6]. Esto se debe a la contribución de genes valiosos para resistencia a enfermedades, insectos y tolerancia a estrés abiótico.

Teniendo presente el contexto y la importancia de estudiar la diversidad genética o variación genética, y su relevancia dentro de CIAT, resulta primordial crear una herramienta que apoye un flujo de trabajo que garantice el uso de los datos producidos por las nuevas tecnologías de secuenciación, la cual permita analizar los datos que se generan, y facilite la obtención, de resultados conforme a la búsqueda de los diferentes tipos de variación presentes en un organismo.

La creación de esta herramienta debe unificar los conceptos biológicos en una herramienta de software que no solo garanticé la gestión de datos NGS si no que reemplace las técnicas actuales poco eficientes en tiempo y coste para una organización dedicada a la investigación en genética y mejora de cultivos.

Actualmente, el uso de herramientas NGS por parte de la comunidad científica de CIAT es poco eficiente ya que llevan a cabo demasiados procesos para acceder a la información que desean utilizar en sus experimentos de laboratorio, generando excesivo consumo de tiempo y desgaste.

Por otra parte, el hecho que estas herramientas bowtie2, Picard y SAMTools se encuentren separadas dificulta muchísimo el orden causando pérdida de datos.

En este sentido, la librería NGSTools es importante en tres aspectos:

1. Calidad de datos.

2. Tiempo de respuesta eficiente.

3. Posibilidad de uso en software y hardware de computadores corrientes.

Un factor determinante para la comunidad científica de CIAT e industria es poder acceder y utilizar esta librería NGSTools de manera fácil y muy intuitiva, que les permita llevar el proceso de análisis de datos NGS sin necesidad de tener muchos conocimientos en otro tipo de estudio diferente al que conocen.

En consecuencia, y después de analizar el poco tiempo para desarrollo requerido por el cliente, los estándares actuales para desarrollo de herramientas NGS, y el conocimiento del desarrollador, se ha llegado a la conclusión que la mejor manera de dar usabilidad a la librería NGSTools bajo una de las siguientes interfaces:

1. Usabilidad mediante la implementación de interfaces Stand-Alone.

2. Usabilidad mediante la implementación de interfaces Web.

3. Usabilidad mediante la implementación de interfaces Plug-in.

DEFINICIÓN DEL CONTEXTO DE NGSTOOLS

En la actualidad con la revolución de la tecnología NGS, se están generando archivos genéticos de tamaños enormes, estos archivos genéticos son indispensables a la hora de realizar cualquier análisis en bioinformática. Se hace preciso crear aplicativos que puedan soportar la carga y almacenamiento de este tipo de archivos. Por ejemplo: los alineamientos de secuencias con respecto a una referencia, es un tipo de proceso llamado mapeo, este proceso utiliza archivos genéticos con gran tamaño y que genera datos de igual o mayor tamaño. Los datos generados por este proceso son vitales en lo demás análisis bioinformáticos como detección de variantes.

En este sentido, las herramientas de código abierto a crear para trabajar con NGS deben poder mantenerse y distribuirse de manera fácil, el mantenimiento de software es la modificación de un producto de software después de la entrega, para corregir errores, mejorar el rendimiento, u otros atributos. Una distribución de software, es un conjunto de software específico ya compilado y configurado que puede ser descargado desde Internet [47].

Conociendo el tipo de investigaciones que se desarrollan en CIAT y las necesidades de la comunidad investigativa con respecto a la creación de nuevas herramientas bioinformáticas. Es indispensable que la futura interfaz gráfica que se integre a la librería NGSTools, soporte la carga de archivos genéticos como los utilizados en CIAT, en ese sentido, es importante que la respuesta del aplicativo para cargar lecturas de frijol, arroz o yuca sea rápida en cuestión de tiempo teniendo presente que este tipo de archivos tiene tamaños mayores a 1 GB.

Otro aspecto importante es, soportar el almacenamiento de estos archivos.
Por otro lado, es necesario garantizar una fácil distribución y mantenimiento de versiones del software cada vez que se genere un cambio nuevo en el aplicativo, esto concebiría alta usabilidad al aplicativo ya que facilita el proceso de instalación para el usuario final, además la corrección o mejora de los procesos ofrecidos dentro los aplicativos algo muy común en la comunidad científica que hace uso de herramientas de código abierto.

Estos elementos son utilizados para definir claramente el contexto sobre el cual
Se va implementar la interfaz gráfica en NGSTools. La anterior descripción se relaciona en la tabla:

Criterio No.	Característica	Nivel
1	Tiempo de carga de archivos genéticos	**Alto**
2	Almacenamiento local de archivos genéticos	**Alto**
3	Distribución y mantenimiento de versiones del software	**Alto**
4	Reutilización de componentes gráficos	**Medio**

Tabla 1 : Características del contexto de implementación de una interfaz gráfica en NGSTools.

1.2.1 COMPARATIVA DE INTERFACES, UTILIZANDO LOS CRITERIOS DEFINIDOS EN LA Tabla 1.

Interfaz Stand-Alone

Criterio 1: Satisface el criterio, debido que al ser la aplicación Stand-Alone, se ejecuta localmente lo que genera que el tiempo de carga de archivos sea bastante rápido.

Criterio 2: Satisface el criterio, debido que al ser la aplicación Stand-Alone, se ejecuta localmente y el almacenamiento de archivos es controlado por el usuario.

Criterio 3: No satisface el criterio, debido que las aplicaciones Stand-Alone, requieren la elaboración de un nuevo archivo jar o ejecutable cada vez que se genere un cambio dentro del aplicativo, obligando al usuario a tener que descargar y reinstalar la aplicación.

Criterio 4: No satisface el criterio, debido que las aplicaciones Stand-Alone, no ofrecen ningún tipo de reutilización de componentes gráficos, por lo cual se obliga al desarrollador a comenzar desde cero la mayor parte de las funcionalidades requeridas por el cliente.

En este sentido, luego de evaluar la implementación de interfaces Stand-Alone, se ha llegado a la conclusión que a pesar de cumplir con los criterios de evaluación uno y dos, se encontró con la dificultad para su distribución en términos de instalación y desarrollo, de esta forma estaría incumpliendo con los criterios de evaluación tres y cuatro. Al no cumplir con los cuatro criterios de evaluación fue descartado el uso de interfaces Stand-Alone.

Interfaz Web

Criterio 1: No satisface el criterio, debido a la dependencia de la red, al ser dependiente de la red la arquitectura web presenta un gran inconveniente para el criterio número uno, en la carga de archivos genéticos que por lo general son de tamaños bastante considerables, hablamos de genomas completamente secuenciados con la tecnología NGS que pueden llegar a tener un tamaño mayor a los 6 Gigabytes (GB), ejemplo el genoma de yuca completamente secuenciado pesa 550,000 Megabytes (MB), las lecturas de arroz pueden pesar 2,5 (GB) y las de frijol 3,5 (GB). Lo que dificulta en gran medida su manejo por la red, al tener el usuario que subir cada archivo con el que va trabajar y posteriormente descargar los archivos que se generan en los diferentes procesos.

Criterio 2: No satisface el criterio, la concepción de aplicativos Web, es mantener en un servidor el repositorio de datos e información, con esta restricción, teniendo en cuenta el número de posibles usuarios y el tamaño de los archivos genéticos hace factible que se presenten fallos en los servidores a la hora de almacenar los archivos con los que trabajen los usuarios del aplicativo.

Criterio 3: Satisface el criterio, las aplicaciones web son de fácil distribución y mantenimiento ya que al estar instaladas en un servidor independiente del usuario final, pueden ser fácilmente actualizados los cambios que se generen y posteriormente montar estos cambios en la web, donde el usuario final puede acceder mediante la web sin necesidad de instalar ni borrar el aplicativo.

Criterio 4: Satisface el criterio, muchos de los actuales generadores de código como Zathura permiten generar y reutilizar componentes gráficos, facilitándole al desarrollador la creación de interfaces graficas desde cero.

Al no cumplir con los primeros dos primeros criterios, se llegó a la conclusión que la implementación de interfaces Web no es factible.

Interfaz Plug-in

Criterio 1: Satisface el criterio, la implementación de un Plug-in en un entorno de trabajo como Eclipse, permite que el usuario maneje locamente los archivos genéticos que va utilizar en el aplicativo, facilitando la carga y generación de archivos.

Criterio 2: Satisface el criterio, la implementación de un Plug-in en un entorno de trabajo como Eclipse, permite que el usuario maneje locamente los archivos genéticos que va utilizar, teniendo presente los recursos y limitantes de hardware.

Criterio 3: Satisface el criterio, debido a que la implementación de un Plug-in, tiene fácil integración a una plataforma de desarrollo como Eclipse. Por otra parte, la fácil distribución que ofrece Eclipse para Plug-in permite mantener un buen control de versiones y de instalación, porque al generar una nueva versión del aplicativo el entorno de trabajo de Eclipse reconoce estas nuevas actualizaciones permitiendo instalarlas de una manera relativamente fácil para usuarios con pocos conceptos de programación.

Criterio 4: Satisface el criterio, la implementación de interfaz Plug-in, permite acceder y modificar funcionalidades ya desarrolladas dentro de la plataforma (Eclipse) ahorrando tiempo en desarrollo y permitiendo cumplir con el criterio número cuatro.

La siguiente tabla muestra el cumplimiento o no de las tres interfaces elegidas para dar usabilidad a la librería NGSTools con respecto a los criterios definidos en la página 17 para evaluar la usabilidad. La calificación obtenida por cada interfaz para determinado criterio es justificada en la pág. 18 y 19.

Criterios de evaluación	Stand-Alone	Web	Plug-in
1	Si	No	Si
2	Si	No	Si
3	No	Si	Si
4	No	Si	Si

Tabla 2: tabla con el resultado de la evaluación de tres Interfaces.

Después, de llevar a cabo esta comparación, se ha tomado la decisión de dar usabilidad a la librería NGSTools mediante la implementación de interfaces Plug-in, se ha denominado como nombre para la interfaz de usuario (GUI) a NGSEP acrónimo del término en inglés (NGSTools Eclipse Plug-in).

Teniendo en cuenta el contexto anterior, a continuación se presentan los objetivos definidos para la realización de este trabajo:

1.3 OBJETIVO GENERAL

Diseñar y construir una interfaz gráfica que proporcione una mejor usabilidad y accesibilidad a la librería NGSTools.

1.3.1 OBJETIVOS ESPECÍFICOS.

- ✓ Definir el contexto para el cual se va implementar la interfaz.

- ✓ Implementar la interfaz gráfica en NGSTools.
- ✓ Evaluar la usabilidad de la interfaz gráfica implementada en NGSTools.

CAPÍTULO 2: ESTADO DEL ARTE Y MARCO TEÓRICO

En este apartado, se explicará los conceptos más relevantes que involucran el desarrollo de la herramienta. De igual forma se expone cada una de las herramientas actuales que cumplen con el mismo flujo de trabajo presentado por la librería NGSTools. Una vez expuesta cada una de las herramientas se presenta una comparativa para determinar que herramienta presenta mayor usabilidad.

2.1 DIVERSIDAD GENÉTICA

"El conocimiento de la diversidad genética de las especies resulta fundamental para diseñar estrategias de conservación adecuadas y desde 1992, es considerado como prioridad en el Programa de Medio Ambiente de las Naciones Unidas" [4]

La diversidad genética juega un papel relevante en el mejoramiento de los cultivos [6]. Esto se debe a la contribución de genes valiosos para resistencia a enfermedades, insectos y tolerancia a estrés abiótico.

2.2 LIBRERÍA NGSTOOLS

Es un marco integrado para el descubrimiento de las variantes genómicas de los datos producidos por NGS. Integra algoritmos desarrollados anteriormente para la detección SNPs, CNV con implementaciones en Java, esta librería fue desarrollada por el Dr. Jorge Duitama Castellanos investigador en bioinformática de CIAT [40].

NGSTools proporciona un modelo de objetos para permitir diferentes tipos de análisis de los datos de secuenciación de alto rendimiento, como se muestra en la Ilustración 1.

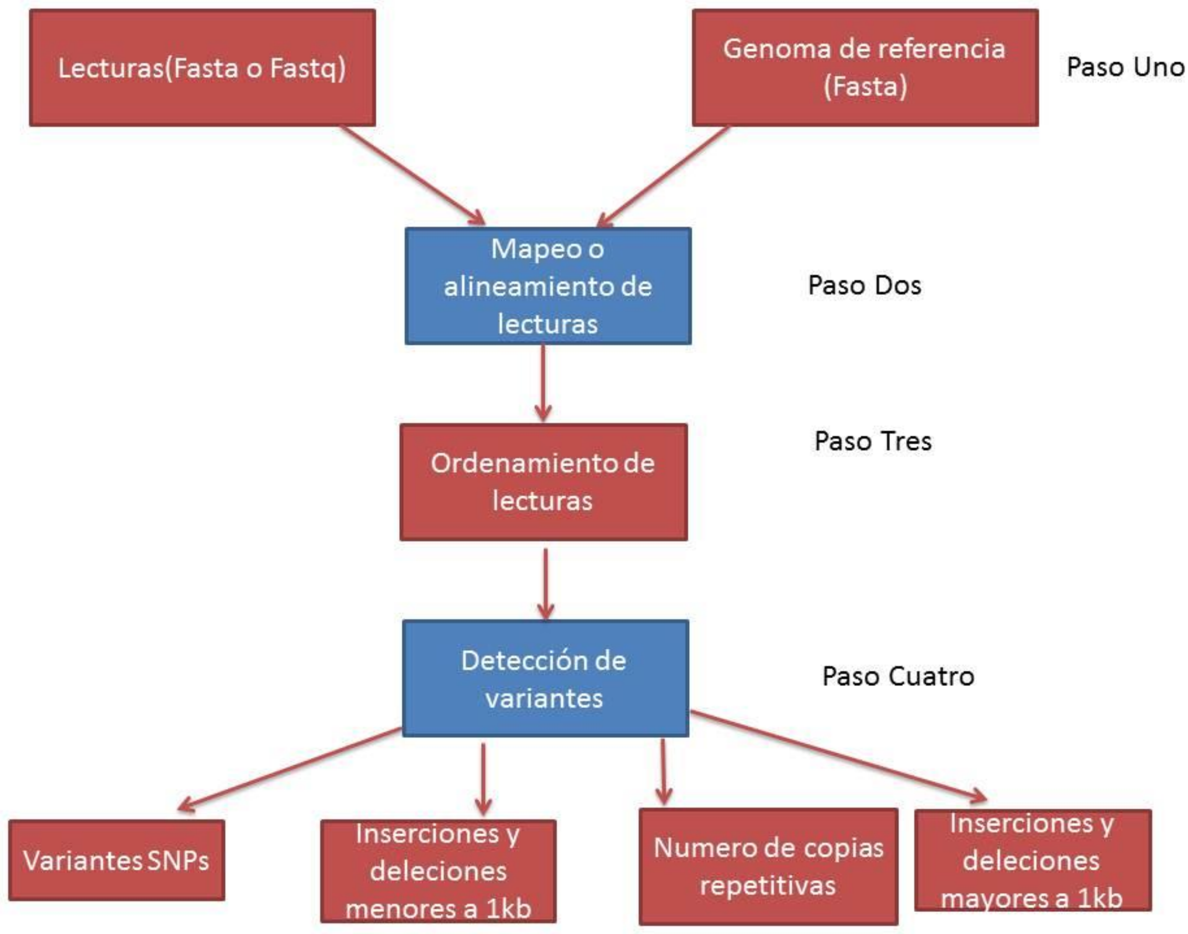


Ilustración 1: Marco de trabajo de la librería NGSTools [40].

El flujo de trabajo de NGSTools, se compone actualmente de las siguientes opciones: Mapeo, Alineamiento de lecturas, Detector de variantes.

- Mapeo: Es el proceso inicial de NGSTools donde se lleva a cabo la comparación entre un genoma de referencia y lecturas de secuencias que son el resultado de secuenciadores como illumina y 454, esta comparación se hace para realizar alineamientos de lecturas respecto al genoma de referencia.

- Alineamiento de lecturas: Es el segundo proceso de NGSTools se lleva a cabo después del mapeo, y se encarga de organizar un archivo SAM producto del proceso anterior (Mapeo).

- Detector de variantes: Es el componente principal de NGSTools, implementa los últimos algoritmos para la detección de SNVs, CNVs, y variantes estructurales. Se enfoca en la comparación de un archivo de entrada (BAM) con las lecturas de un genoma producto de secuenciadores como illumina, Sanger®, 454, contra un genoma de referencia, con el fin de encontrar variantes genómicas [40].

NGSTools también proporciona utilidades para calcular estadísticas de calidad y cobertura, lo que facilita llevar a cabo la anotación funcional de variantes.

“El formato elegido para procesar las alineaciones de todos los componentes de NGSTools es SAM (o BAM), que permite integrar NGSTools con programas de mapeo de uso común como bowtie2” [7].

La siguiente Ilustración 2, muestra el catálogo de variantes genómicas de las cuales NGSTools es capaz de detectar en uno de sus procesos.

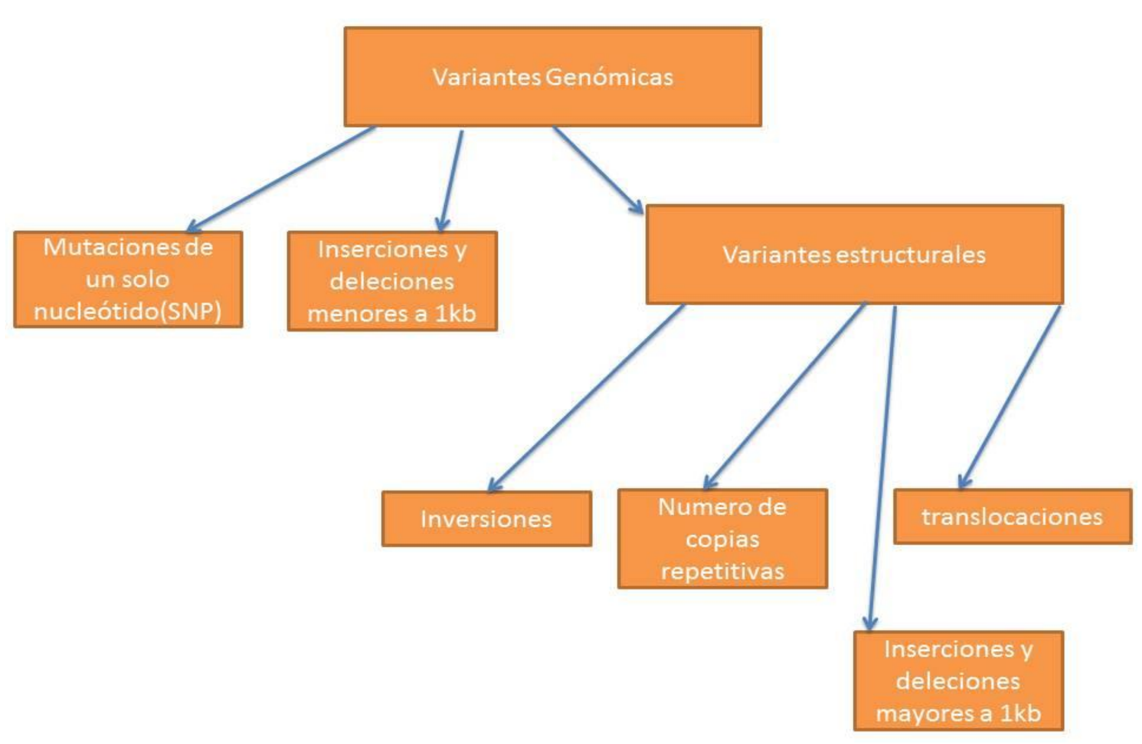


Ilustración 2: Catálogo de variantes estructurales [46].

2.3 SNPs

Es un tipo de variación genómica estructural, acrónimo del término en inglés *Single Nucleotide Polymorphism,* polimorfismo de un nucleótido único, es la forma más sencilla de mutación genética, ya que consisten en el cambio de un soló nucleótido en una secuencia, la Ilustración 3, es un ejemplo de este tipo de variante genómica. Su distribución es de manera heterogénea a lo largo del genoma y se encuentran en regiones codificantes de proteínas denominadas exones como en la no codificantes que son los intrones.

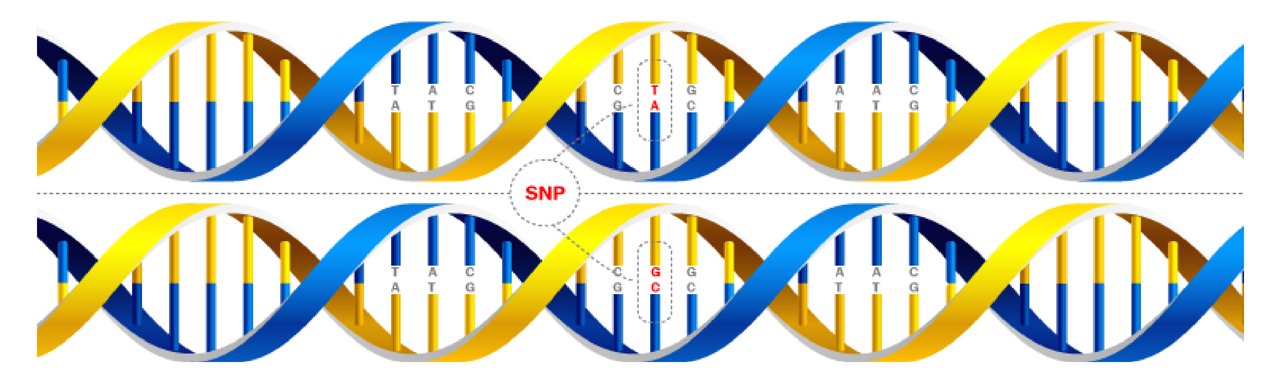


Ilustración 3: SNP cambio de un nucleótido de la hebra amarilla Tiamina por guanina y cambio de nucleótido en la hebra azul de Adenina a Citosina [11].

2.4 INDEL (INSERCIONES Y DELECIONES DE NUCLEOTIDOS)

Es un tipo de variación genómica, son las ganancias o pérdidas de nucleótidos en la secuencia del ADN (una inserción es la adición de uno o varios nucleótidos y la deleción es la perdida de uno o varios nucleótidos) como se muestra en la Ilustración 4.

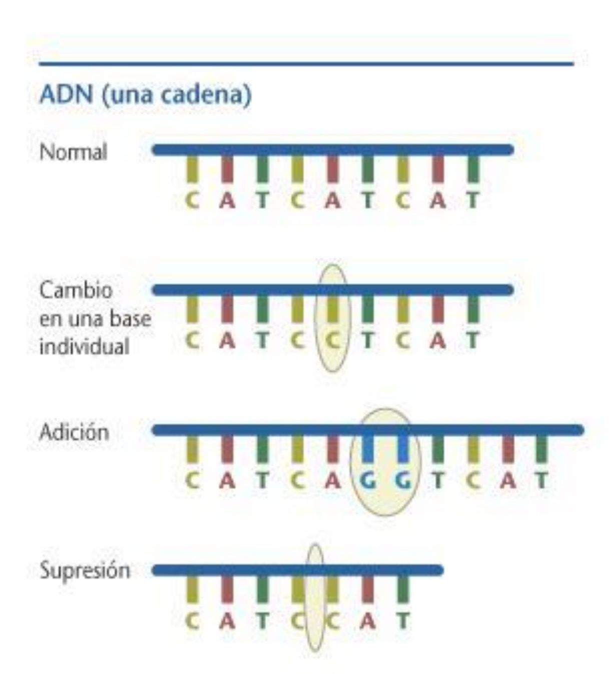


Ilustración 4: Ejemplo de variaciones genómicas en una cadena de ADN, se pueden apreciar las siguientes variaciones: SNP, inserción o adición, deleción o supresión [12].

2.5 CNV (VARIANTES DE NÚMERO DE COPIA)

Una variación del número de copia (CNV) es cuando el número de copias de un gen en particular varía de un individuo a otro.

‘’Es una forma de variación estructural, son alteraciones del ADN de un genoma que se traduce en la célula que tienen un número anormal de copias de una o más secciones de la ADN. CNV corresponden a las relativamente grandes regiones del genoma que se han eliminado (menos de la cantidad normal) o duplicados (más que el número normal) en ciertos cromosomas’’ [34]. La Ilustración 5, es un ejemplo de los diversos tipos de CNV que pueden presentarse en el ADN.

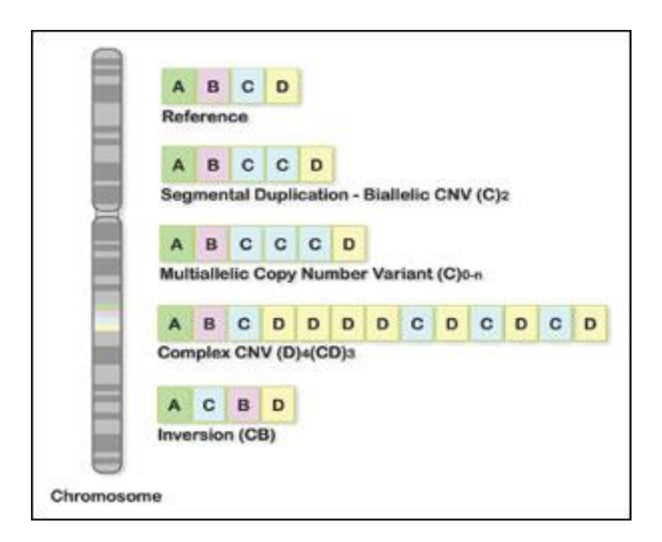


Ilustración 5: Variación CNV en la secuencia ABCD de un locus de un cromosoma [34].

2.6 USABILIDAD

La usabilidad en un producto de software : Es La capacidad que tiene dicho producto para ser atractivo, entendido, aprendido y usado por el usuario cuando es utilizado bajo unas condiciones específicas [48].

En este sentido, la usabilidad está directamente relacionada con la satisfacción de un cliente que adquiere un producto de software. Esto indica que un sistema usable debe poseer los atributos: capacidad de aprendizaje, eficiencia en el uso, facilidad de memorizar, tolerante a errores y satisfactorio [8].

Por otra parte, el autor del libro de usabilidad no me hagas pensar de Steve Krug afirma que, para que un software sea fácil de utilizar o tenga alta usabilidad; no se trata de "que nada importante esté a más de dos clics de distancia", de "hablar el lenguaje del usuario" o, incluso, de "ser coherente". Se trata de entender que es y cómo funciona una herramienta sin necesidad de agotar esfuerzos pensando en ella [41].

Steve Krug, expresa en su libro un conjunto de normas para evaluar si una aplicación web tiene usabilidad. El conjunto de normas son:

1. No me hagas pensar: En una aplicativo web, cualquier cosa puede detenernos y hacernos pensar innecesariamente. Por ejemplo, los nombres de las cosas. Los típicos culpables son los nombres bonitos o ingeniosos, los producidos por el departamento de

marketing, los nombres específicos de la empresa y los nombres técnicos que no nos son familiares.

2. Todo no se puede hacer obvio: El objetivo para cada aplicativo web debería ser que fuera evidente, que el usuario final con tal sólo con mirar supiera de lo que se trata y la forma de usarse. No obstante, algunas veces, si hace algo en concreto que es realmente original o innovador, o incluso, algo bastante complicado, debe conformarse con la claridad. En una página fácil de entender hay que pensar incluso un poquito para entenderla. La apariencia de las cosas, sus nombres bien escogidos, la disposición de la página y los textos pequeños y cuidadosamente elaborados tienen que funcionar bien en su conjunto para conseguir un reconocimiento casi instantáneo.

3. ¿Por qué es todo esto tan importante?: En ocasiones esto es cierto, pero se sorprendería al conocer el tiempo que algunas personas tardan en abandonar los sitios que les frustran. Muchas personas que se encuentran con problemas en un sitio tienden a culparse a sí mismas y no al propio sitio [41].

Con este conjunto de normas Steve Krug propone evaluar la usabilidad de una interfaz gráfica de un aplicativo web.

En este sentido, a continuación se presenta la definición de cada una de las heurísticas propuestas para evaluar la usabilidad del diseño de interfaz gráfica de usuario del autor Jakob Nielsen [8].

10 heurísticas de usabilidad para el diseño de interfaz de usuario

Resumen: Los 10 principios más generales para el diseño de interacción. Se les llama "heurística", ya que están más en la naturaleza de las reglas generales de las directrices de usabilidad específicos.

Visibilidad del estado del sistema

El sistema siempre debe mantener a los usuarios informados sobre lo que está pasando, a través de información adecuada en un plazo razonable.

Correspondencia entre el sistema y el mundo real

El sistema debe hablar el idioma de los usuarios, con palabras, frases y conceptos familiares para el usuario, en lugar de términos orientados a sistemas. Siga las convenciones del mundo real, haciendo que la información aparezca en un orden natural y lógico.

Control y libertad del usuario

Los usuarios a menudo eligen funciones del sistema por error y necesitarán salidas de emergencia para salir del estado no deseado, sin tener que pasar por un diálogo extendido. Soporte de deshacer y rehacer.

Consistencia y estándares

Los usuarios no deberían tener que preguntarse si diferentes palabras, situaciones o acciones significan lo mismo. Siga las convenciones de la plataforma.

Prevención de errores

Incluso mejor que buenos mensajes de error, es un diseño cuidadoso que evita que un problema se produzca en primer lugar, minimizando los riesgos de que puedan ocurrir. Se debe realizar un buen diseño de mensajes de error que den la posibilidad al usuario de retraerse antes de que se realice la acción y se comprometan los datos.

Reconocer antes que recordar

Minimizar la carga de memoria del usuario mediante objetos de decisiones, acciones y opciones visibles. El usuario no debería tener que recordar información de una parte de un diálogo a otro. Las instrucciones de uso del sistema deben ser visibles o fácilmente accesibles cuando sea apropiado.

La flexibilidad y eficiencia de uso

El sistema se debe diseñar para que lo puedan manejar diferentes tipos de usuarios, en función de su experiencia con la aplicación. De esta manera se aumentará la productividad del usuario y se ganará en usabilidad. Permitiendo a los usuarios adaptarse a las acciones frecuentes.

Diseño estético y minimalista

Los diálogos no deben contener información que es irrelevante para la tarea que está realizando el usuario. Cada unidad adicional de información en un diálogo compite con las unidades relevantes de información y disminuye su visibilidad relativa.

Ayude a los usuarios a reconocer, diagnosticar y recuperación de errores

Los mensajes de error deben ser expresados en un lenguaje sencillo (sin códigos), indicar con precisión el problema y sugerir una solución constructiva.

Ayuda y documentación

A pesar de que es mejor si el sistema puede ser utilizado sin la documentación, puede ser necesario proporcionar al usuario ayuda y documentación. Dicha información debe ser fácil de buscar, enfocada en la tarea del usuario. Se deben listar sólo los pasos necesarios para la realización de la tarea.

La IEEE define un conjunto de reglas para que los aplicativos tengan alta usabilidad, este conjunto de reglas son definidos de esta manera [43]:

Advertencia del estado y/o retroalimentación
La aplicación debe mostrar los indicadores de estado o de alerta cuando no se cumplan ciertas condiciones dentro de la aplicación.

Perfil de usuario
Cada usuario debería ser capaz de establecer varios parámetros que controlan la ejecución de un proceso.

Agregación de comando
El usuario debe ser capaz de invocar un archivo con una colección de comandos en él y ejecutarlos.

Recuperación de un fallo
Cuando el sistema, el procesador, o la red fallen, el usuario debe estar en la capacidad de no perder ningún trabajo.

Apoyo al usuario internacional
El usuario debe tener capacidad de utilizar el sistema en un idioma y un formato de pantalla que sea familiar.

Ayuda
El usuario debe poder acceder a documentación de la herramienta que le permite comprender mejor las diferentes pantallas.

Después de analizar las definiciones de los criterios Jakob Nielsen, Steve Krug y la IEEE. Se pueden obtener los principios básicos en los que se afirma la usabilidad:

- ✓ Facilidad de Aprendizaje.
- ✓ Flexibilidad.
- ✓ Robustez.

En la implementación de usabilidad a un producto de software, aporta importantes beneficios referentes a los costes de desarrollo, la calidad del producto y la satisfacción del cliente.

Además de estos beneficios se encuentran:

- ✓ Incremento del uso de la aplicación
- ✓ Reducción de los costes de soporte a la aplicación ya que resulta un producto fácil de instalar, de aprender y de usar.
- ✓ Reducción de los costes de mantenimiento de la aplicación.

- ✓ Aumento de la calidad de desarrollo de la organización con la adquisición de buenas prácticas a base de la aplicación de la usabilidad en todo el proceso de desarrollo lo que redundará en el desarrollo de futuros proyectos.

- ✓ Aumento de la satisfacción del cliente reduciendo el esfuerzo de uso por parte del usuario y mejorando la calidad de vida de los usuarios.

2.7 HERRAMIENTAS QUE TRABAJAN CON NGS

En este apartado, se lleva a cabo la explicación y comparación de usabilidad de interfaces entre herramientas que utilizan la tecnología NGS o secuenciación de alto de rendimiento y que tienen un flujo de trabajo similar a NGSTools.

Para llevar a cabo esta comparación, se utilizan las heurísticas propuestas por Jakob Nielsen uno de los padres de la usabilidad. Se elige utilizar estas heurísticas por qué a pesar que Jakob Nielsen las describe en su libro indicando que son para medir la usabilidad de interfaces Web, estas heurísticas aplican para cualquier sistema, al cual se deseé medir que tan fácil es de usar para un usuario final.

Nielsen a comparación de las reglas establecidas por Steve Krug y el método USAP de la IEEE, muestra un mejor balance entre aspectos generales del sistema y detalles específicos de la interfaz. Estableciendo un conjunto de heurísticas que aseguran el cumplimiento total del contexto de las herramientas bioinformáticas requerido para evaluar la usabilidad. De acuerdo a la definición de usabilidad del apartado anterior, la usabilidad es considerada un atributo de calidad importante a la hora de desarrollar y utilizar un sistema de software.

A continuación y teniendo como referente las diez heurísticas propuestas por Jakob Nielsen y expuestas en el libro “Usability Engineering” se propone una evaluación para calificar la usabilidad de las herramientas [8].

En este sentido, la siguiente Tabla 3 describe las herramientas actuales que utilizan secuenciación de alto rendimiento y detectan variantes estructurales con igual flujo de trabajo a NSGTools.

Nombre Herramienta	Sistema Operativo	Archivos de entrada	Otros archivos de entrada	Archivos de salida	Variantes Identificadas
GATK	Linux, Mac	BAM/SAM		VCF.	SNP, INDEL
SAMtools	Linux, Mac, Windows	BAM/SAM	Fasta	VCF.	SNP, INDEL
SNVer	Linux, Mac, Windows	BAM/SAM		VCF, BAM, SAM.	SNP, INDEL
NGSEP	Linux, Mac, Windows	BAM/SAM	Fasta, Fastq	VCF, BAM, SAM.	SNP, INDEL

Tabla 3: Herramientas que trabajan con datos de NGS y tienen igual flujo de trabajo o pipeline.

2.8 HERRAMIENTAS RELACIONADAS

En este apartado, se describe el estado del arte de cada una de las herramientas que se asemejan a NGSTools, posteriormente se realiza una comparación de usabilidad de interfaces entre estas herramientas y se genera un análisis de resultados. A partir de la Ilustración 6, se puede resaltar que las herramientas GATK (UnifiedGenotyper), SAMtools y SNVer tienen un flujo de trabajo similar al que ofrece NGSTools, puesto que reciben como entrada un archivo SAM/BAM y generan archivos de salida en formato VCF, al igual que ofrecen detención de variantes genéticas como SNPs e Indeles.

2.8.1 GATK (UnifiedGenotyper)

GATK, es una herramienta que permite identificar variantes genómicas del ADN de un organismo, en su descripción dice: “digamos que usted tiene diez exones y desea identificar las mutaciones que todos tienen en común con GATK puede hacer eso. Asimismo, permite saber, qué mutaciones son específicas de un grupo de pacientes, en comparación con una cohorte sana. En GATK puede hacer eso también. De hecho, la GATK es el estándar de la industria para este tipo de análisis” [13].

“Debido a la forma en que está construido, la GATK es muy genérico y se puede aplicar a todos los tipos de conjuntos de datos y a los problemas de análisis de genomas. Se puede utilizar para el descubrimiento, así como para la validación. En sólo exones puede tener manipulación tan feliz como con genomas completos”.

A pesar que GATK en un principio fue creado para ser utilizado con el genoma humano, se puede utilizar para realizar de detección de variantes SNPs, Indel en cualquier secuencia de cualquier organismo, generando datos importantes [13].

2.8.1.1 Plataforma y requisitos para la instalación de GATK

GATK está diseñado para ejecutarse en Linux y Mac Os X. En el caso de Windows se puede instalar mediante Cygwin aunque no lo recomiendan ya que no pueden dar ningún tipo de apoyo específico para este sistema operativo. Se dice que en un futuro estará disponible para Android.

2.8.1.2 Interfaz

GATK no tiene una interfaz gráfica de Usuario, se accede a todas sus herramientas mediante línea de comandos, según la página de GATK para hacer uso de esta herramienta no se necesita ningún tipo de conocimiento en programación [13].

2.8.1.3 Comando estructura y los argumentos de la herramienta

Todas las herramientas del flujo de trabajo de GATK, se llaman utilizando un comando de fácil acceso por ejemplo: el siguiente comando cuenta el número de lecturas en una secuencia y genera un archivo BAM con el resultado.

```
java-jar GenomeAnalysisTK.jar \
    -T CountReads \
    -R example_reference.fasta \
    -I example_reads.bam
```

El argumento de java–jar invoca el motor GATK, y el argumento de la –T le dice a la herramienta qué desea ejecutar.

Argumentos como -R para la referencia del genoma y -I para el archivo de entrada también se dan al motor GATK y se puede utilizar con todas las herramientas del mismo modo.

2.8.1.4 El flujo de trabajo básico

El flujo de trabajo o pipeline de GATK se compone de: el mapeo inicial (Mapping), refinamiento de lecturas iniciales (Aligned Reads), detención de Indel y SNP con una o varias lecturas, y por ultimo recalibración a nivel de calidad para variantes. Estos pasos son los mismos que se especifican para resecuenciación [13].

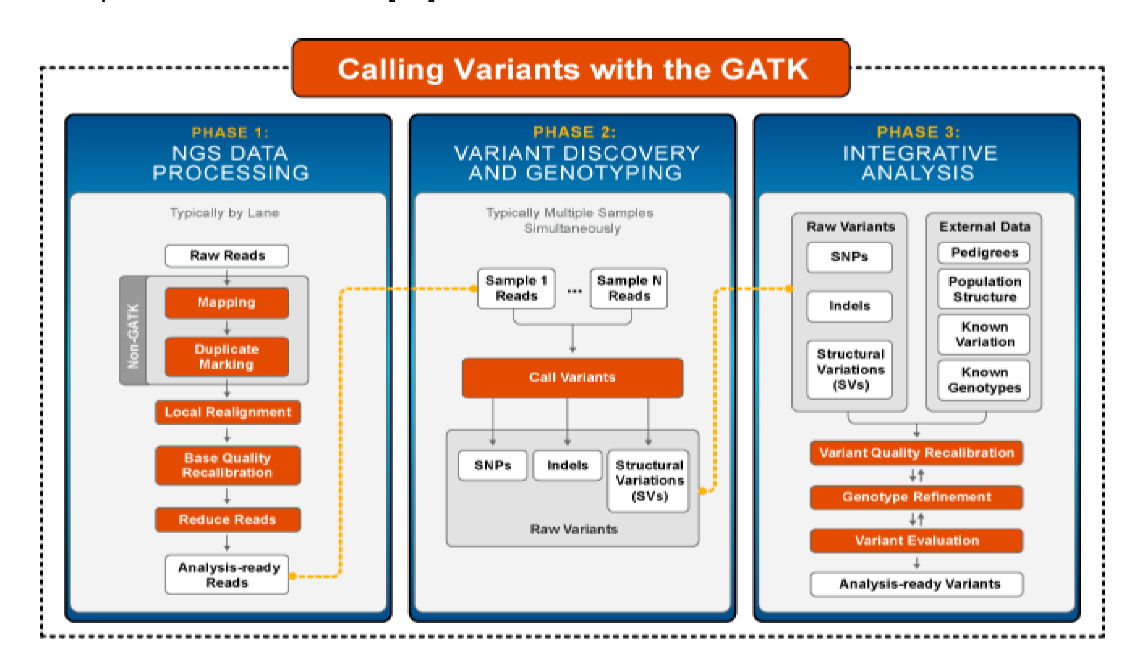


Ilustración 6: Flujo de trabajo de GATK [13].

2.8.2 SAMTOOLS

Es una herramienta que trabaja con datos NGS y que permite detectar variantes como: SNPs e indeles pequeños. El flujo de trabajo ofrecido por SAMTOOLS se describe en la Ilustración 7, Según la descripción de la página del proyecto SAMTOOLS es: "una herramienta que proporciona varias utilidades para la manipulación de las alineaciones en el formato SAM, incluida la selección, la fusión, la indexación y la generación de alineaciones en un formato por cada posición" [14].

En ese sentido, el formato Sam por descripción explicita de los creadores de SAMTOOLS es un formato genérico para almacenar grandes alineamientos de secuencias de nucleótidos. SAM tiene como objetivo ser un formato que:

- ✓ Sea lo suficientemente flexible como para almacenar toda la información generada por los programas de alineación como bowtie2.
- ✓ Sea lo suficientemente simple para ser fácilmente generado por los programas de adaptación o conversión de formatos de alineación existentes.
- ✓ Compacto en tamaño del archivo.
- ✓ Permita que el archivo que se indexa sea por posición genómica para recuperar de manera eficiente todas las lecturas, para su posterior alineación a un locus [14].

2.8.2.1 Interfaz

SAMTOOLS no cuenta con interfaz gráfica de usuario. El acceso a todas sus herramientas es por línea de comandos. Actualmente se encuentra disponible para el sistema operativo Linux, para Windows también aunque no ofrece mayor documentación para su uso en este sistema operativo.

Todas las herramientas de SAMTOOLS se llaman utilizando la misma estructura básica de comando. He aquí un ejemplo sencillo que detecta SNP e Indeles de una secuencia de levadura:

```
Samtools        mpileup         -uf

/home/juan/workspace/TestPlugin/src/sacCer_SGD_refgenome_20110301.fa

/home/juan/Desktop/ Samtools/Unselected_bowtie2_sorted2.bam | bcftools view -vcg - >

/home/juan/Desktop/Samtools/UnselectSamtools.vcf
```

El argumento "Samtools" se utiliza para acceder a las funcionalidades de SAMTOOLS, mpileup sirve para utilizar el flujo de trabajo para detectar variantes genómicas, -uf se utiliza para referenciar la ruta que contiene el genoma de referencia, luego se deja un espacio y se ingresa

la ruta donde se encuentra la secuencia que se desea comparar, la opción bcftools view -vcg -> se utiliza para generar el archivo de salida producto de la comparación en un formato vcf.

La Ilustración 7, muestra el pipeline o flujo de trabajo que ofrece SAMtools para hacer detención de variantes.

2.8.2.2 Flujo de trabajo de SAMtools

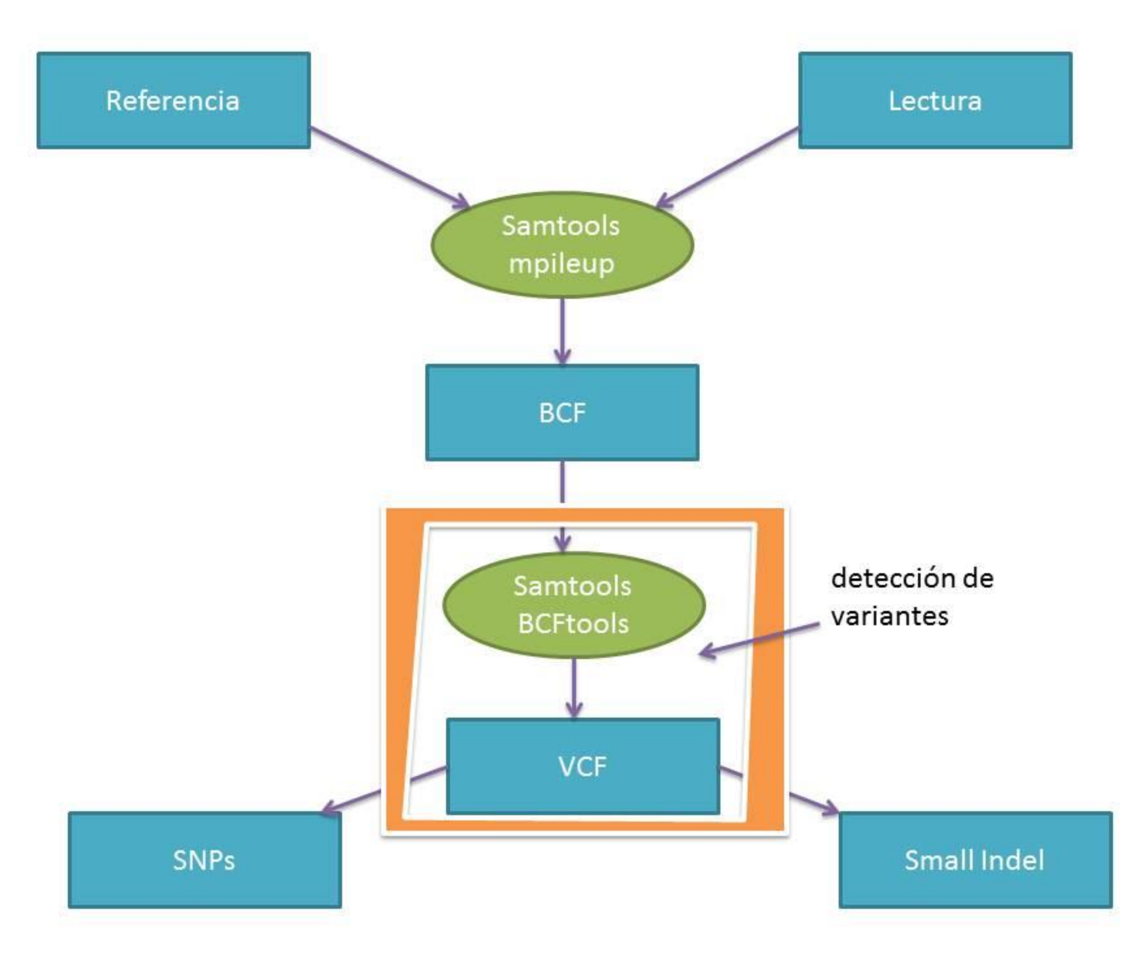


Ilustración 7: Marco de trabajo o pipeline de Samtools para detectar variantes [14].

2.8.3 SNVer (Single Nucleotide Variants Caller)

SNVer es una herramienta estadística para detectar a las variantes genómicas SNPs e indeles pequeños en el análisis de muestras individuales o multi-muestras.

Según la descripción de la página del proyecto: "SNVer corre muy rápido, por lo que es factible para el análisis de datos de secuenciación de todo el genoma de un organismo secuenciado, es una de las pocas herramientas existentes que son capaces de detectar variantes (tanto la variación de un solo nucleótido SNP como la de pequeños indeles)".

Hay más herramientas existentes de NGS para realizar detección de variantes (2-5) incluyendo SNVer, sin embargo, se basan en una interfaz de línea de comandos. Los usuarios deben ejecutar comandos no interactivos para el funcionamiento de estos programas seguidos de análisis [15].

2.8.3.1 Interfaz

SNVer cuanta con una interfaz gráfica de usuario llamada SNVerGUI, mediante la cual se puede acceder a la detección de variantes genómicas después de configurar varios parámetros [15].

SNVer se encuentra disponible para los siguientes sistemas operativos: Windows (win32/x86_64), Windows (win32/x86), Mac OSX (cocoa/x86_64), Mac OSX (cocoa/x86), Linux (gtk/x86_64), Linux (gtk/x86).

En la Ilustración 8, se puede visualizar la interfaz gráfica de la aplicación SNVer.

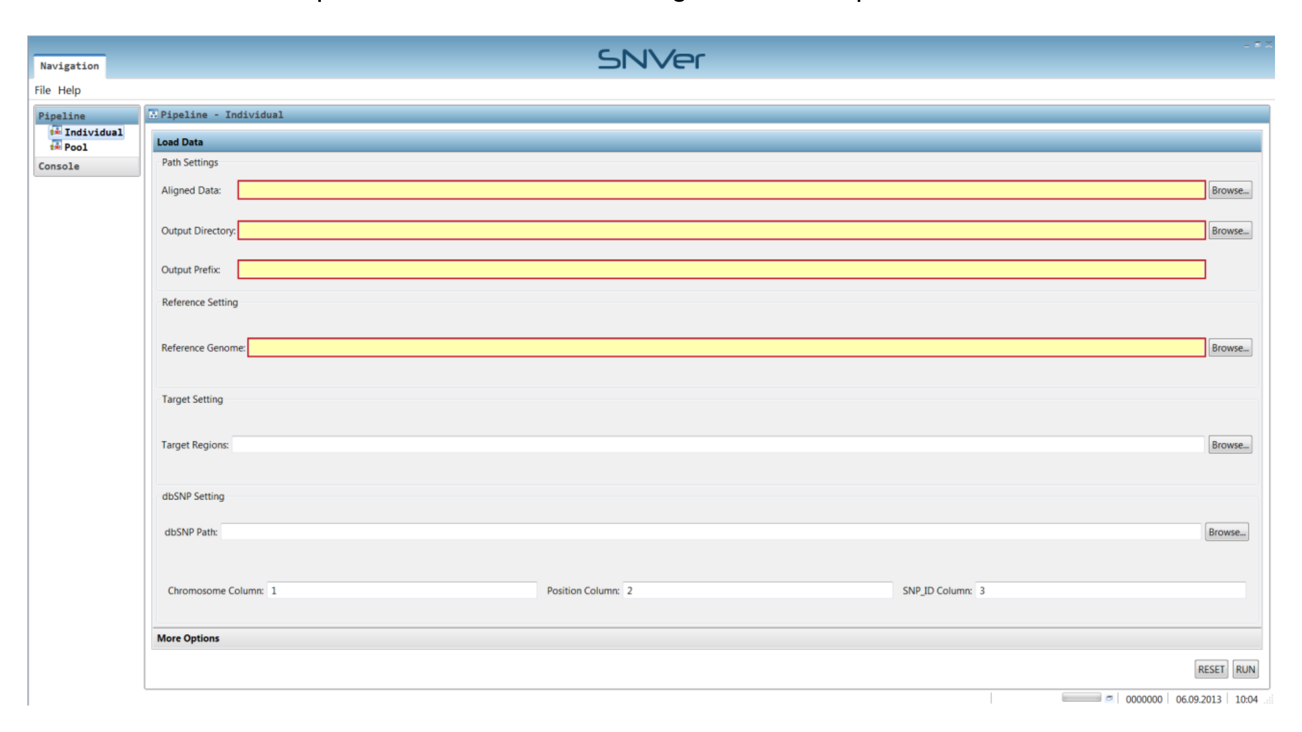


Ilustración 8: Interfaz Gráfica de usuario de SNVer [15].

2.8.3.2 Flujo de trabajo de SNVer

El flujo de trabajo presentado por SNVerGUI en análisis de datos NGS puede utilizar un pool o datos individuales de secuenciación, la Ilustración 9 muestra el pipeline ofrecido por SNVer.

1. Mapeo de lecturas.
2. Eliminación de datos duplicados.

3. Detección de SNPs.
4. Anotación de genes con respecto a las variantes encontradas.

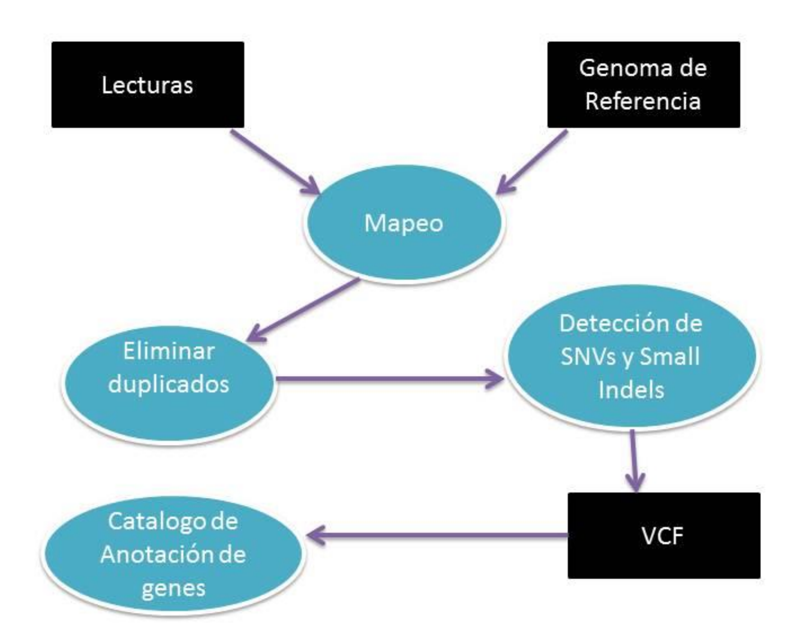


Ilustración 9: Pipeline o flujo de trabajo de SNVerGUI.

2.9 ESCALA A UTILIZAR PARA CALIFICAR LA USABILIDAD DE GATK, SAMTOOLS, SNVER.

De acuerdo al contexto explicado en el capítulo uno y las herramientas descritas en los párrafos anteriores, se propone una escala para evaluar los criterios descritos para calificar la usabilidad propuestos por Jakob Nielsen´s, en ese sentido los criterios elegidos para calificar la usabilidad de estas cuatro herramientas de acuerdo al contexto de bioinformática son:

- ✓ Visibilidad del estado del sistema
- ✓ Control y libertad del usuario
- ✓ Prevención de errores
- ✓ Correspondencia entre el sistema y el mundo real
- ✓ Reconocer antes que recordar
- ✓ Estética y diseño minimalista
- ✓ Ayudar a los usuarios a reconocer, diagnosticar y recuperación de errores
- ✓ Ayuda y documentación

A continuación, se crea una tabla con escalas para evaluar los 8 criterios establecidos de acuerdo al contexto de bioinformática descritos en el párrafo anterior.

Calificación	Porcentaje	Descripción
1	20%	El criterio evaluado no cumple con la especificación descrita para este.
2	40%	El criterio evaluado cumple levemente con la especificación descrita para este.
3	60%	El criterio evaluado cumple parcialmente lo descrito en la especificación para este.
4	80%	El criterio evaluado cumple con faltas menores en la especificación descrita para este.
5	100%	El criterio se cumple completamente con la especificación descrita para este.

Tabla 4: Escala para evaluar la usabilidad de la (GUI).

Para utilizar la tabla propuesta se debe calificar el criterio o heurística conforme a la columna calificación de la tabla de Escala de evaluación para heurísticas de usabilidad, para elegir una calificación se debe leer la fila correspondiente a la columna descripción que más se asemeje al valor que cree el evaluador que corresponde para dicho criterio, de esta manera el valor tomado representara un porcentaje, descrito en la tabla, este porcentaje representa un valor del total del criterio o heurística a calificar.

La fórmula para calcular el porcentaje de cumplimiento de una heurística o criterio es:

Valor de cumplimiento=valor calificado*peso heurística/Total Efectividad;

Donde cada una de las variables significa:

Valor calificado: corresponde al valor de calificación proporcionado por la tabla de evaluación para heurísticas de usabilidad, que se toma de acuerdo a la descripción que más se ajuste al estado actual de la herramienta con respecto a esta heurística, este valor representa un porcentaje de total para heurística.

Peso heurísticas: corresponde al peso de cada una de las heurísticas justificada en el párrafo anterior. Este valor equivale a dividir cien sobre ocho que son la cantidad de heurísticas a evaluar, el valor es 12.5.

Total Efectividad: corresponde al porcentaje total de efectividad de la heurística.

Valor de cumplimiento: es el valor total que corresponde al porcentaje de cumplimiento de la heurística calificada.

2.10 COMPARATIVA DE HERRAMIENTAS.

En la Tabla 5 y Tabla 6, se pretende comparar las herramientas GATK, SAMtools y SNVer con el fin de conocer cuál de todas estas herramientas tiene mejor usabilidad para un usuario final.

Las calificaciones para la evaluación, fueron otorgadas por un usuario con pleno conocimiento del contexto de herramientas bioinformáticas. Para una mayor apreciación de las calificaciones obtenidas por SNVer, la única de las tres herramientas con interfaz gráfica con la cual se va comparar NGSEP, se elabora una muestra con imágenes del proceso de detección de variantes genómicas de SNVer.

Herramienta (SNVer)

Heurística: Visibilidad del estado del sistema

Pregunta: ¿La aplicación mantiene siempre informado al usuario del estado del sistema, así como de los caminos que este pueda tomar con una retroalimentación visual apropiada en tiempo razonable?



Ilustración 10: Pantalla de SNVer para detectar SNPs e Indeles.

Una vez ingresados los datos correspondientes como se puede observar en la Ilustración 10, automáticamente la aplicación abre la pantalla de consola y muestra al usuario una retroalimentación visual de las entradas para el proceso y de los archivos que se van a generar Ilustración 13, Ilustración 14, así mismo como el estado de la ejecución del proceso mediante una barra de progreso Ilustración 12 y la impresión de información relevante del proceso Ilustración 11.

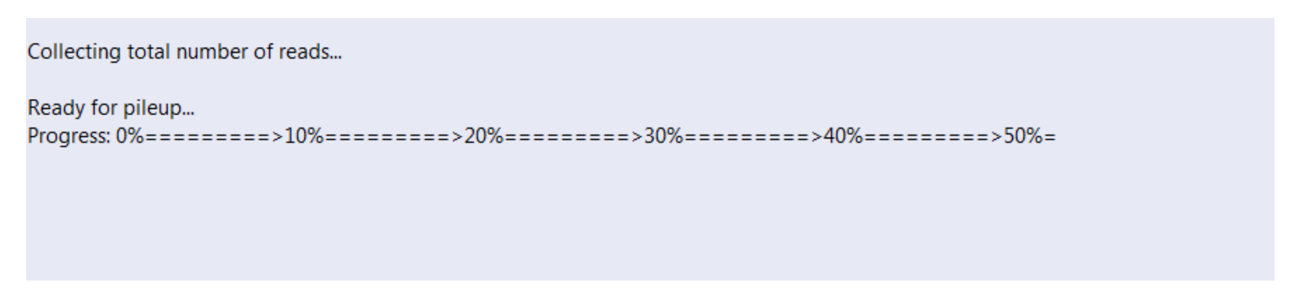


Ilustración 11: Impresión de SNVer en pantalla del estado actual de proceso ejecutado.

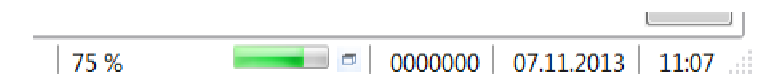


Ilustración 12: Barra de progreso generada por SNVer, marca que porcentaje de progreso se ha ejecutado.

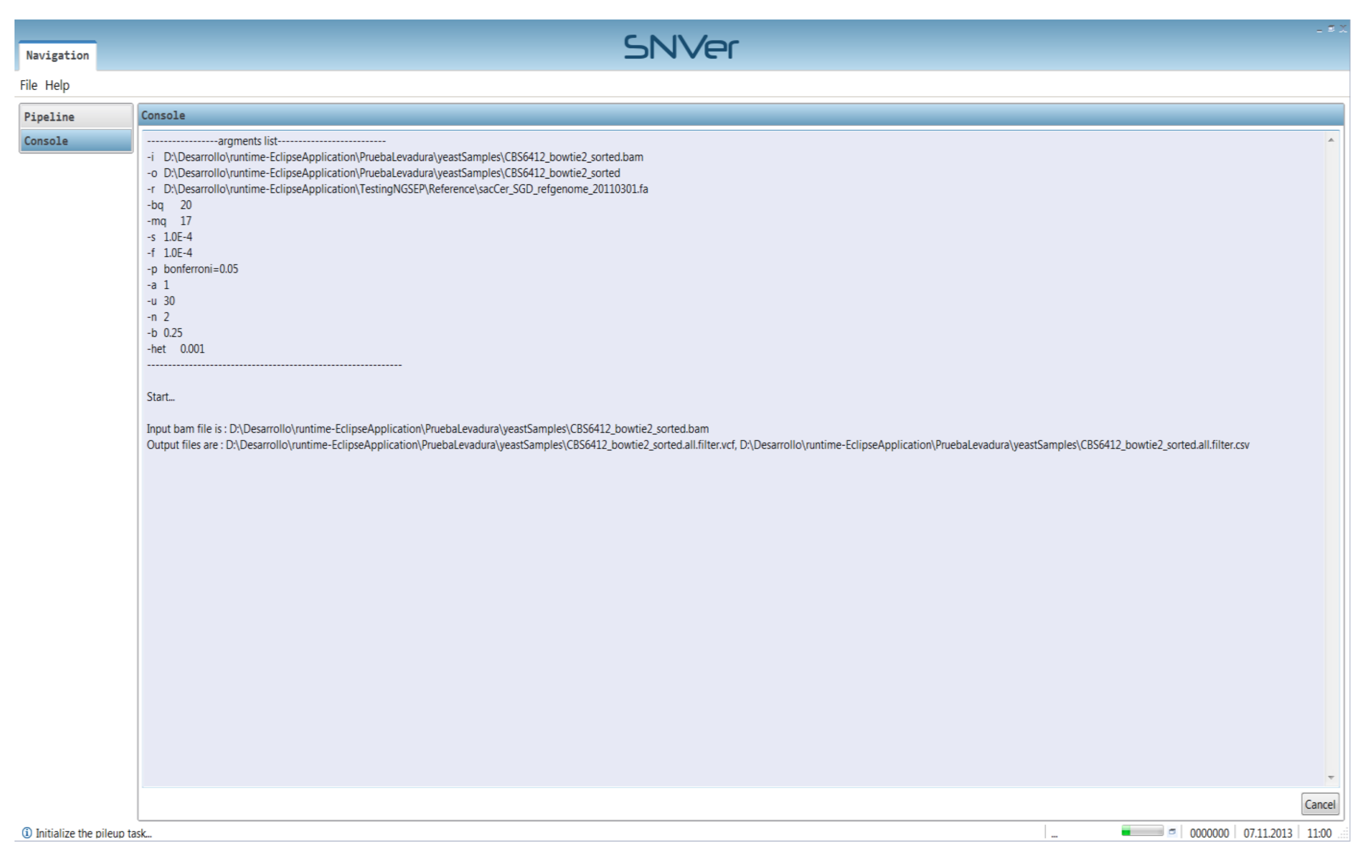


Ilustración 13: Pantalla de SNVer con información relevante del proceso de detección de variantes.

Result@D:\Desarrollo\runtime-EclipseApplication\PruebaLevadura\yeastSamples\CBS6412_bowtie2_sorted

CHROM	POS	ID	REF	ALT	QUAL	FILTER	DP	AC	FS	SP	PV
chrI	174	.	C	-T	.	PASS	11	11	-1.0	-1.0	5.645029E-...
chrI	177	.	G	C	.	PASS	11	11	1.0	0.006	9.570426E-...
chrI	201	.	A	C	.	PASS	9	9	1.0	0.02	8.4107725E...
chrI	209	.	T	C	.	PASS	9	8	1.0	0.035	1.7488753E...
chrI	210	.	C	A	.	PASS	9	8	1.0	0.035	1.8148089E...
chrI	220	.	T	+C	.	PASS	13	6	-1.0	-1.0	1.9223467E...
chrI	220	.	T	C	.	PASS	13	8	0.491	0.035	2.5996194E...
chrI	245	.	C	+T	.	PASS	12	10	-1.0	-1.0	1.05101034...
chrI	249	.	T	C	.	PASS	12	12	1.0	0.019	5.0113525E...
chrI	250	.	G	A	.	PASS	12	11	1.0	0.033	1.8599909E...
chrI	286	.	A	T	.	PASS	14	14	1.0	0.029	3.521312E-...
chrI	458	.	C	-A	.	PASS	19	13	-1.0	-1.0	1.4093227E...
chrI	476	.	G	T	.	PASS	25	16	1.0	0.002	7.4365754E...
chrI	481	.	C	-CACTT	.	PASS	28	9	-1.0	-1.0	1.9705674E...
chrI	485	.	T	C	.	PASS	28	19	1.0	0.0	9.527E-41
chrI	509	.	G	A	.	PASS	70	69	1.0	0.0	0.0
chrI	518	.	A	-C	.	PASS	98	37	-1.0	-1.0	4.0104316E...
chrI	519	.	C	T	.	PASS	101	43	0.133	0.0	0.0
chrI	521	.	A	G	.	PASS	108	49	0.038	0.0	0.0
chrI	531	.	C	T	.	PASS	117	58	0.002	0.0	0.0
chrI	558	.	C	T	.	PASS	128	63	0.0	0.0	1.110223E-...
chrI	562	.	C	T	.	PASS	132	52	0.208	0.002	0.0
chrI	573	.	G	T	.	PASS	124	59	0.002	0.0	1.110223E-...
chrI	588	.	C	A	.	PASS	127	125	1.0	0.429	0.0
chrI	594	.	T	C	.	PASS	132	56	0.358	0.114	0.0
chrI	595	.	C	G	.	PASS	127	125	1.0	0.0	0.0
chrI	610	.	G	A	.	PASS	134	134	1.0	0.0	0.0
chrI	633	.	T	C	.	PASS	141	78	0.0	0.286	0.0
chrI	681	.	G	T	.	PASS	72	46	0.002	0.151	0.0
chrI	688	.	A	C	.	PASS	62	44	0.005	0.048	0.0
chrI	693	.	T	A	.	PASS	52	52	1.0	0.063	0.0
chrI	694	.	A	T	.	PASS	52	39	0.0	0.0	0.0
chrI	701	.	C	A	.	PASS	57	57	1.0	0.002	0.0

Functional Annotation with wANNOVAR

Ilustración 14: Archivo de salida automáticamente desplegado en la pantalla una vez terminado el proceso.

Calificación obtenida: la máxima calificación, cumple con la heurística de manera acorde a la pregunta realizada, calificación igual a 5.

Heurística: Control y libertad del usuario

Pregunta:

¿La interfaz de la aplicación permite controlar la iteración de los procesos, de esta manera dejando el control de la aplicación al usuario y permitiéndole interactuar con los elementos contenidos en la pantalla?

La interfaz de SNVer permite controlar la iteración de cada proceso, mediante botones como los de correr el aplicativo o cancelar la ejecución Ilustración 15, Ilustración 17 . También ofrece la posibilidad de visualizar al usuario una serie de pantallas pertenecientes al proceso una vez este corriendo Ilustración 16, permitiendo al usuario interactuar con los elementos que contiene las pantallas.

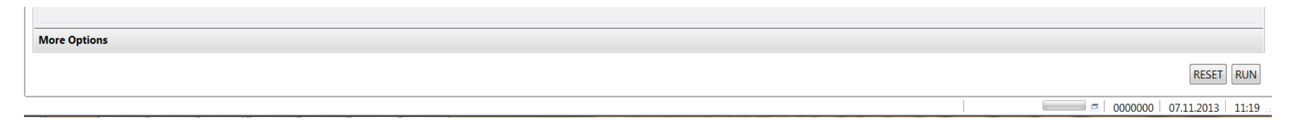


Ilustración 15: botones dentro de la interfaz gráfica de SNVer para cancelar y arrancar el proceso de detección de variantes.

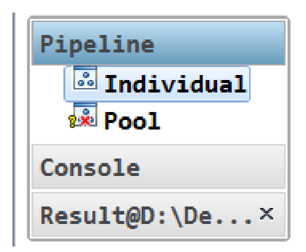


Ilustración 16: Pantallas que se pueden visualizar dentro de SNVer cuando un proceso esté en iteración.



Ilustración 17: Botón para cancelar el proceso de detección de variantes de SNVer en la pantalla.

Calificación obtenida: la máxima calificación, cumple con la heurística de manera acorde a la pregunta realizada, calificación igual a 5.

Heurística: Correspondencia entre el sistema y el mundo real

Pregunta: ¿La interfaz muestra mensajes en el idioma del usuario, cuando se habla de idioma se refiere a palabras, frases y conceptos familiares para el usuario, siempre en el contexto de la aplicación?

Pantalla de SNVer con mensajes relevantes del proceso de detección de variantes Ilustración 18, estos mensajes se muestran en un idioma familiar al usuario entendiéndose que es biólogo o bioinformático.

```
Console
----------------argments list-------------------------
-i   D:\Desarrollo\runtime-EclipseApplication\PruebaLevadura\yeastSamples\CBS6412_bowtie2_sorted.bam
-o   D:\Desarrollo\runtime-EclipseApplication\PruebaLevadura\yeastSamples\CBS6412_bowtie2_sorted
-r   D:\Desarrollo\runtime-EclipseApplication\TestingNGSEP\Reference\sacCer_SGD_refgenome_20110301.fa
-bq     20
-mq     17
-s   1.0E-4
-f   1.0E-4
-p   bonferroni=0.05
-a   1
-u   30
-n   2
-b   0.25
-het     0.001
-----------------------------------------------------------

Start...

Input bam file is : D:\Desarrollo\runtime-EclipseApplication\PruebaLevadura\yeastSamples\CBS6412_bowtie2_sorted.bam
Output files are : D:\Desarrollo\runtime-EclipseApplication\PruebaLevadura\yeastSamples\CBS6412_bowtie2_sorted.all.filter.vcf, D:\Desarrollo\runtime-EclipseApplication\PruebaLevadura\yeastSamples\CBS6412_bowtie2_sorted.all.filter.csv

Collecting total number of reads...

Ready for pileup...
Progress: 0%=========>10%=========>20%=========>30%=========>40%=========>50%=========>60%=========>70%=========>80%=========>90%=========>100%

11559389 SNVs and 25793 indels have been tested
Filtering variants based on Bonferroni correction at 0.05...

Time usage is 598 seconds
Done!
```

Ilustración 18: Información relevante del proceso de detección de variantes de SNVer en ejecución.

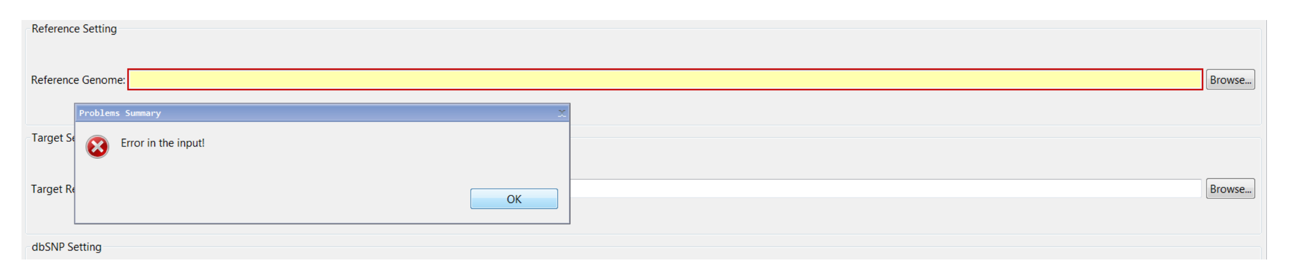


Ilustración 19: Mensaje de excepción en un capo de entrada de la pantalla de detección de variantes de SNVer.

Si el usuario no ingreso o ingreso mal un parámetro la pantalla genera mensajes de excepción marcando donde fue el error en un idioma entendible para el usuario final Ilustración 19.

Calificación obtenida: la máxima calificación, cumple con la heurística de manera acorde a la pregunta realizada, calificación igual a 5.

Heurística: Reconocer antes que recordar

Pregunta: ¿El diseño de la interfaz permite reducir la carga de memoria para un usuario final, se refiere a que si la interfaz ayuda al usuario a no tener que recordar información para ir de un proceso a otro a la hora de realizar una iteración?

SNVer se compone únicamente de dos procesos detección de variantes y anotación de genes a partir de un archivo vcf Ilustración 20. En este sentido, SNVer muestra de manera independiente las pantallas que pertenecen a los procesos mencionados, el proceso de anotación de genes se abre en el navegador de internet explorer por defecto Ilustración 21, lo cual genera problemas a la hora de recordar información importante del proceso de detección de variantes, puesto que el usuario debe ir del aplicativo al navegador y viceversa para ver la información que necesita recordar.

Result@D:\Desarrollo\runtime-EclipseApplication\PruebaLevadura\yeastSamples\CBS6412_bowtie2_sorted

CHROM	POS	ID	REF	ALT	QUAL	FILTER	DP	AC	FS	SP	PV
chrI	174	.	C	-T	.	PASS	11	11	-1.0	-1.0	5.645029E-...
chrI	177	.	G	C	.	PASS	11	11	1.0	0.006	9.570426E-...
chrI	201	.	A	C	.	PASS	9	9	1.0	0.02	8.4107725E...
chrI	209	.	T	C	.	PASS	9	8	1.0	0.035	1.7488753E...
chrI	210	.	C	A	.	PASS	9	8	1.0	0.035	1.8148089E...
chrI	220	.	T	+C	.	PASS	13	6	-1.0	-1.0	1.9223467E...
chrI	220	.	T	C	.	PASS	13	8	0.491	0.035	2.5996194E...
chrI	245	.	C	+T	.	PASS	12	10	-1.0	-1.0	1.05101034...
chrI	249	.	T	C	.	PASS	12	12	1.0	0.019	5.0113525E...
chrI	250	.	G	A	.	PASS	12	11	1.0	0.033	1.8599909E...
chrI	286	.	A	T	.	PASS	14	14	1.0	0.029	3.521312E-...
chrI	458	.	C	-A	.	PASS	19	13	-1.0	-1.0	1.4093227E...
chrI	476	.	G	T	.	PASS	25	16	1.0	0.002	7.4365754E...
chrI	481	.	C	-CACTT	.	PASS	28	9	-1.0	-1.0	1.9705674E...
chrI	485	.	T	C	.	PASS	28	19	1.0	0.0	9.527E-41
chrI	509	.	G	A	.	PASS	70	69	1.0	0.0	0.0
chrI	518	.	A	-C	.	PASS	98	37	-1.0	-1.0	4.0104316E...
chrI	519	.	C	T	.	PASS	101	43	0.133	0.0	0.0
chrI	521	.	A	G	.	PASS	108	49	0.038	0.0	0.0
chrI	531	.	C	T	.	PASS	117	58	0.002	0.0	0.0
chrI	558	.	C	T	.	PASS	128	63	0.0	0.0	1.110223E-...
chrI	562	.	C	T	.	PASS	132	52	0.208	0.002	0.0
chrI	573	.	G	T	.	PASS	124	59	0.002	0.0	1.110223E-...
chrI	588	.	C	A	.	PASS	127	125	1.0	0.429	0.0
chrI	594	.	T	C	.	PASS	132	56	0.358	0.114	0.0
chrI	595	.	C	G	.	PASS	127	125	1.0	0.0	0.0
chrI	610	.	G	A	.	PASS	134	134	1.0	0.0	0.0
chrI	633	.	T	C	.	PASS	141	78	0.0	0.286	0.0
chrI	681	.	G	T	.	PASS	72	46	0.002	0.151	0.0
chrI	688	.	A	C	.	PASS	62	44	0.005	0.048	0.0
chrI	693	.	T	A	.	PASS	52	52	1.0	0.063	0.0
chrI	694	.	A	T	.	PASS	52	39	0.0	0.0	0.0
chrI	701	.	C	A	.	PASS	57	57	1.0	0.002	0.0

Functional Annotation with wANNOVAR

Ilustración 20: botón para acceder al proceso de anotación e genes a partir de la finalización del proceso de detección de variantes.

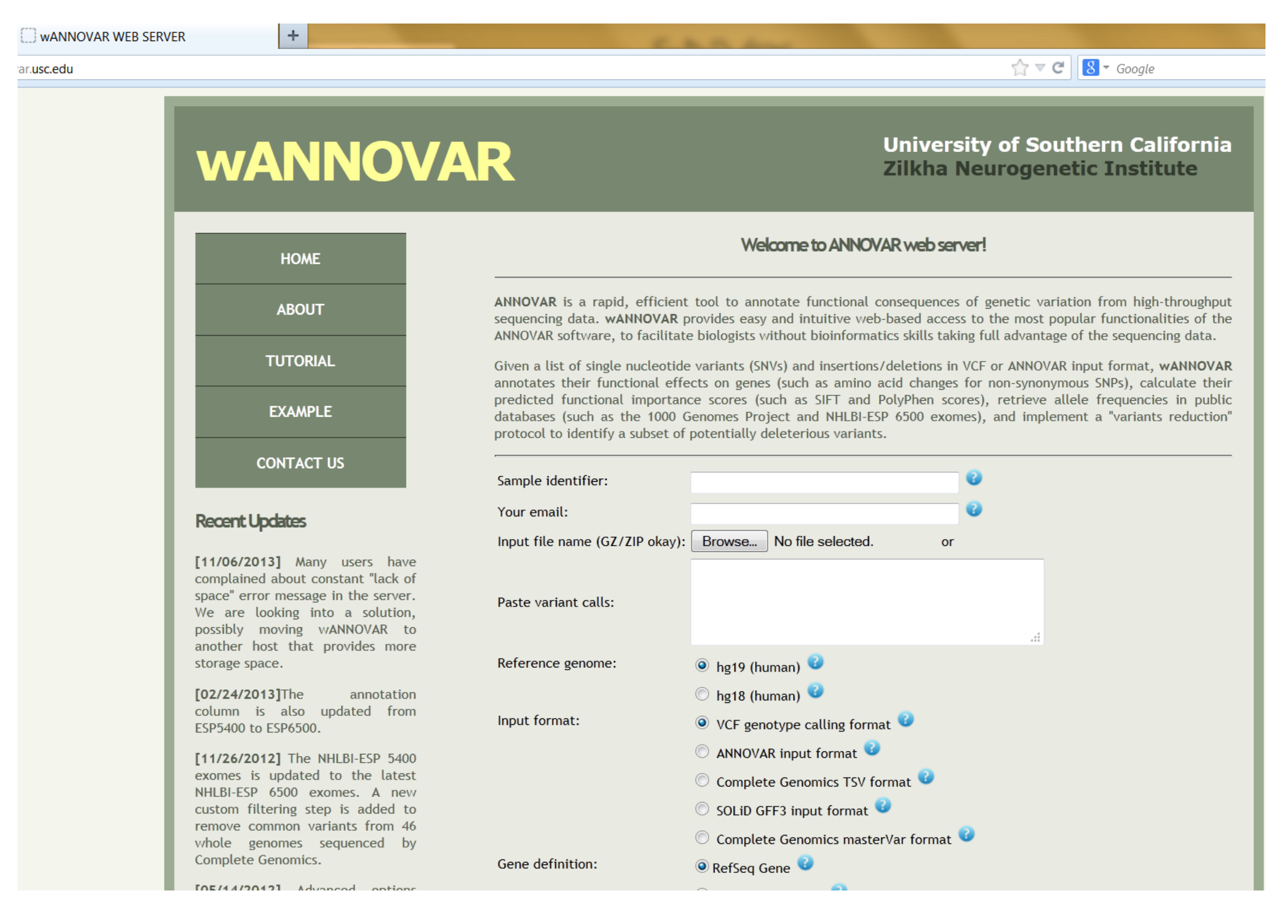


Ilustración 21: Pantalla proceso anotación de genes de SNVer.

Calificación obtenida: la heurística evaluada cumple parcialmente lo descrito en la pregunta para realizada, la calificación es igual a 3.

Heurística: Prevención de errores

Pregunta: ¿La Aplicación tiene un buen diseño de mensajes de error que den la posibilidad al usuario de retraerse antes de que se realice la acción y se comprometan los datos?

Si el usuario no ingreso o ingreso mal un parámetro la pantalla genera mensajes de excepción marcando donde fue el error en un idioma entendible para el usuario final y que permite parar el proceso, si el usuario lo desea Ilustración 19.

Calificación obtenida: la máxima calificación, cumple con la heurística de manera acorde a la pregunta realizada, calificación igual a 5.

Heurística: Estética y diseño minimalista

Pregunta: ¿Los mensajes de la aplicación contienen información relevante para la tarea que está realizando el usuario, por otro lado el diseño de la interfaz es simple, fácil de aprender, fácil de usar y con fácil acceso a las funcionalidades que ofrece la aplicación?

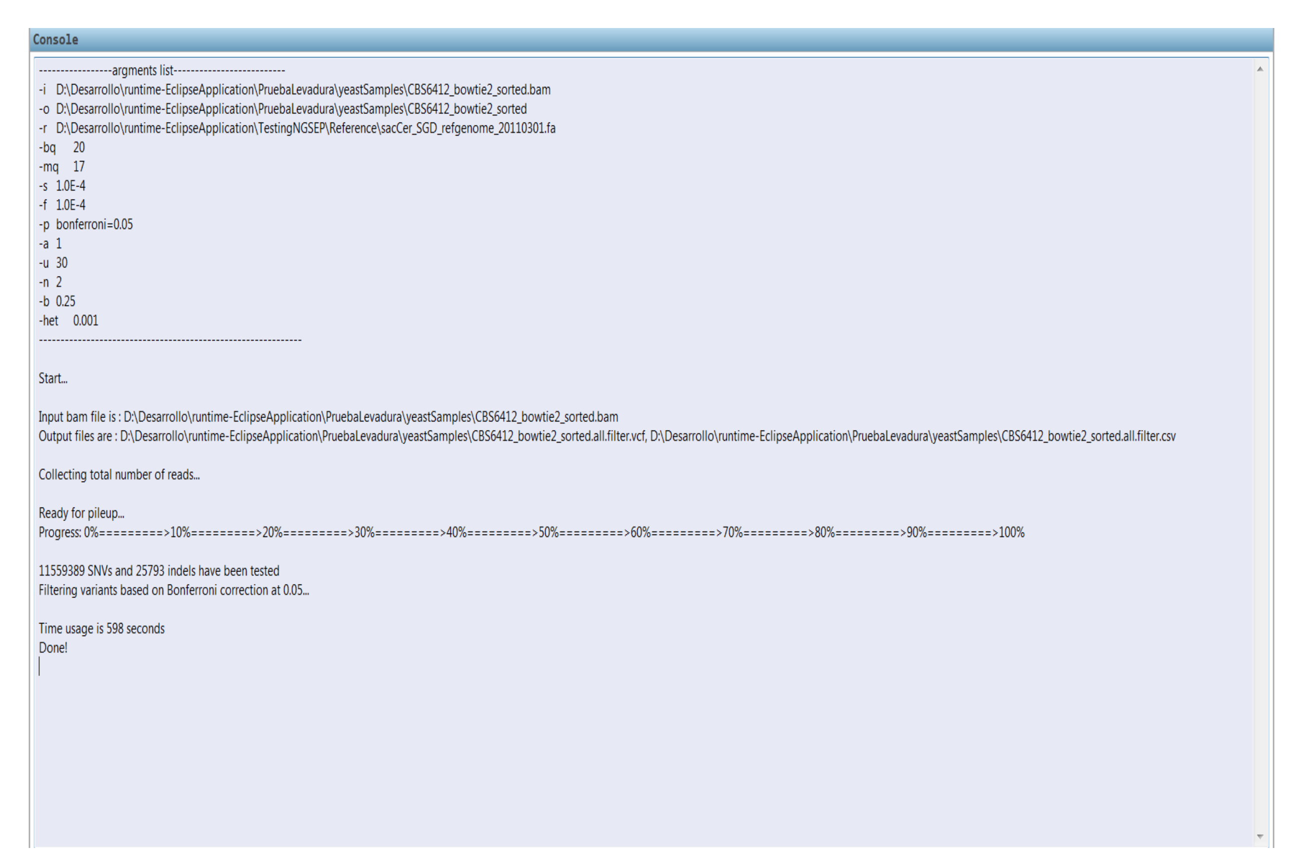


Ilustración 22: Mensajes de la SNVer respecto a la ejecución del proceso de detección de variantes.

Las Ilustración 22 y muestran como la pantalla del proceso de detección de variantes de SNVer tiene una interfaz simple ya que, tiene entradas con títulos de tamaños grandes y claros a la vista del usuario, además de contener botones acordes para arrancar el proceso y para cancelarlo, además posee un menú lateral que permite navegar dentro de las otras pantallas que contiene SNVer. La generación de mensajes con respecto a la información de los procesos es concisa y con información importante del estado actual del proceso.

Calificación obtenida: la máxima calificación, cumple con la heurística de manera acorde a la pregunta realizada, calificación igual a 5.

Heurística: Ayudar a los usuarios a reconocer, diagnosticar y recuperar errores

Pregunta: ¿La aplicación tiene mensajes de error en lenguaje entendible por el usuario y sin código de lenguajes de programación, los mensajes indican el error y sugieren como solucionarlo?

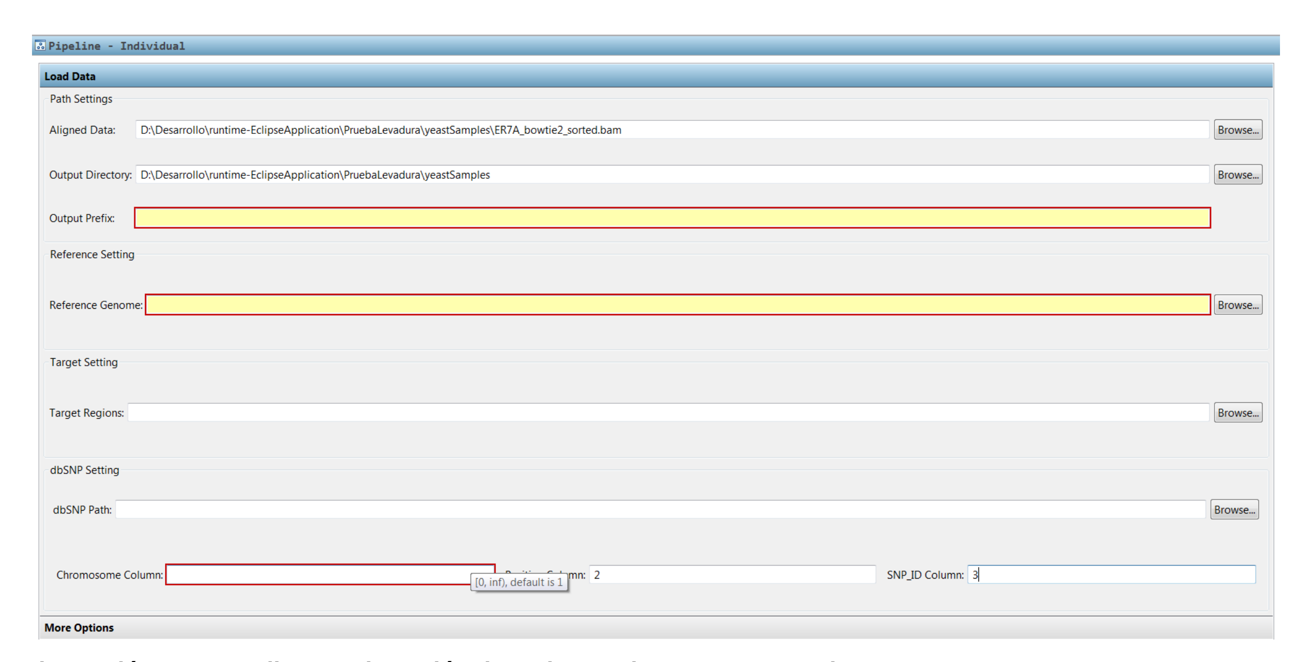


Ilustración 23: Pantalla para detección de variantes de SNVer marcando errores.

SNVer genera mensajes de error en un lenguaje entendible para el usuario final, marca las caja de texto donde se genere el error por los datos ingresados del usuario Ilustración 23, Ilustración 19: Mensaje de excepción en un capo de entrada de la pantalla de detección de variantes de SNVer., además de crear un dialogo de mensajes de error que indica que hay errores en una entrada de la pantalla, sin embargo, SNVer no genera ningún tipo de comentario acerca de cómo solucionar el error.

Calificación obtenida: La heurística evaluada cumple con faltas menores en la pregunta realizada, calificación igual a 4.

Heurística: Ayuda y documentación

Pregunta: ¿La aplicación tiene manual de usuario, la información es fácil de encontrar y enfocada a la tarea que el usuario realiza, se listan los pasos necesarios para la realización de la tarea?

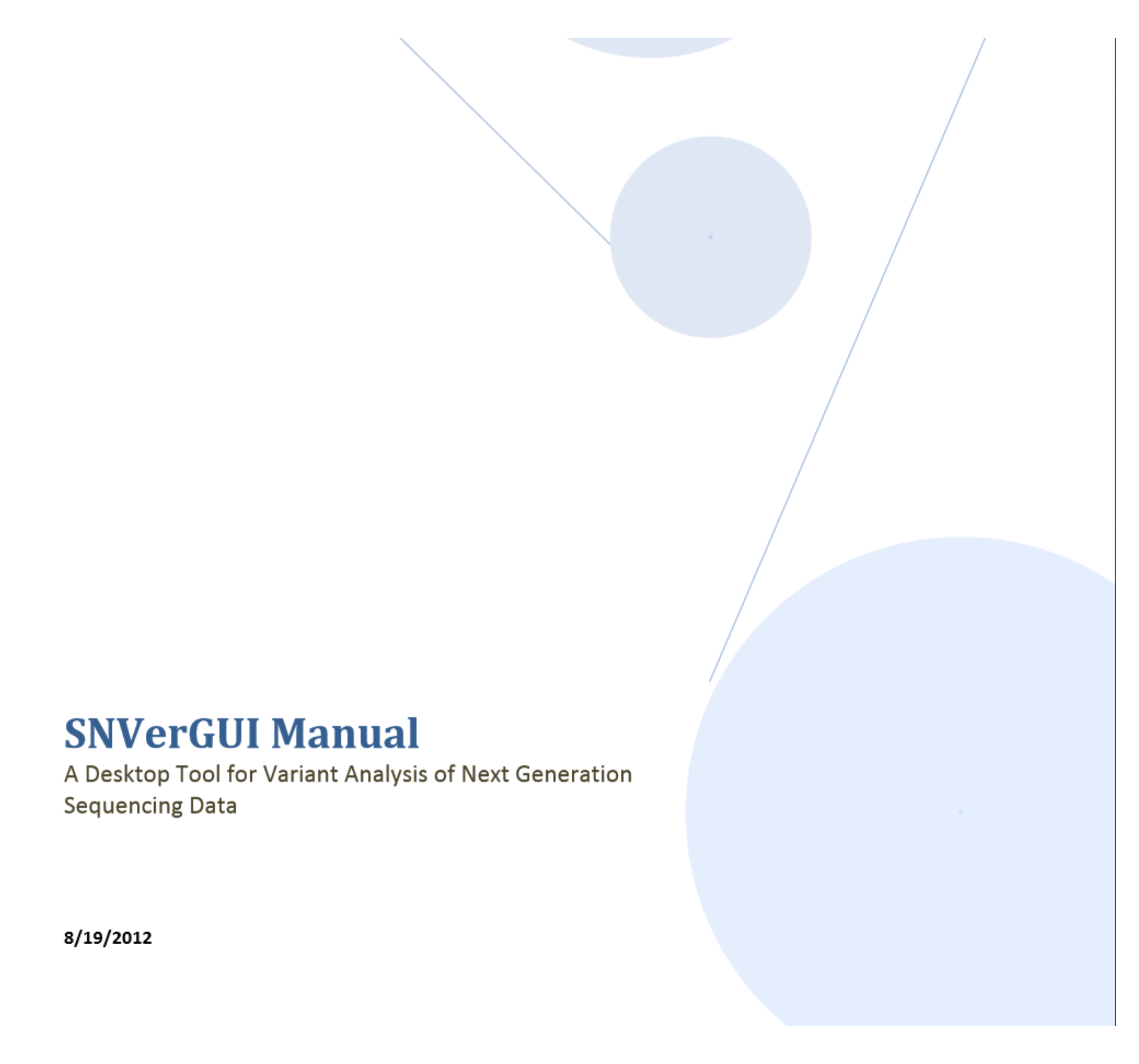


Ilustración 24: Manual de usuario de SNVer.

El manual de usuario de SNVer Ilustración 24, ofrece un contenido de fácil navegación, también la posibilidad mediante links de ir al capítulo que el usuario desea Ilustración 25, además explica en cada capítulo o sección del manual detalladamente para que sirve cada proceso y cada entrada que dato recibe.

Content

Ilustración 25: Índice del manual de usuario de SNVer.

Calificación obtenida: la máxima calificación, cumple con la heurística de manera acorde a la pregunta realizada, calificación igual a 5.

La Tabla 5 contiene las calificaciones otorgadas para las tres herramientas por un usuario con pleno conocimiento del contexto de herramientas bioinformáticas.

Heurística	Pregunta	GATK	SAMtools	SNVer
Visibilidad del estado del sistema	¿La aplicación mantiene siempre informado al usuario del estado del sistema, así como de los caminos que este pueda tomar con una retroalimentación visual apropiada en tiempo razonable?	**3**	**3**	**5**
Control y libertad del usuario	¿La interfaz de la aplicación permite controlar la iteración de los procesos, de esta manera dejando el control de la aplicación al usuario y permitiéndole interactuar con los elementos contenidos en la pantalla?	**1**	**1**	**5**
Correspondencia entre el sistema y el mundo real	¿La interfaz muestra mensajes en el idioma del usuario, cuando se habla de idioma se refiere a palabras, frases y conceptos familiares para el usuario, siempre en el contexto de la aplicación?	**5**	**4**	**5**
Reconocer antes que recordar	¿El diseño de la interfaz permite reducir la carga de memoria para un usuario final, se refiere a que si la interfaz ayuda al usuario a no tener que recordar información para ir de un proceso a otro a la hora de realizar una iteración?	**1**	**1**	**3**
Prevención de errores	¿La Aplicación tiene realizar un buen diseño de mensajes de error que den la posibilidad al usuario de retraerse antes de que se realice la acción y se comprometan los datos?	**5**	**5**	**5**
Estética y diseño minimalista	¿Los mensajes de la aplicación contienen información relevante para la tarea que está realizando el usuario, por otro lado el diseño de la interfaz es simple, fácil de aprender, fácil de usar y con fácil acceso a las funcionalidades que ofrece la aplicación?	**3**	**3**	**5**
Ayudar a los usuarios a reconocer, diagnosticar y recuperar errores	¿La aplicación tiene mensajes de error en lenguaje entendible por el usuario y sin código de lenguajes de programación, los mensajes indican el error y sugieren como solucionarlo?	**5**	**4**	**4**
Ayuda y documentación	¿La aplicación tiene manual de usuario, la información es fácil de encontrar y enfocada a la tarea que el usuario realiza, se listan los pasos necesarios para la realización de la tarea?	**5**	**5**	**5**

Tabla 5: Evaluación realizada con la escala de la Tabla 4 aplicada a las herramientas GATK, SAMtools y SNVer.

Heurística	Pregunta	GATK	SAMtools	SNVer
Visibilidad del estado del sistema	¿La aplicación mantiene siempre informado al usuario del estado del sistema, así como de los caminos que este pueda tomar con una retroalimentación visual apropiada en tiempo razonable?	**7.5**	**7.5**	**12.5**
Control y libertad del usuario	¿La interfaz de la aplicación permite controlar la iteración de los procesos, de esta manera dejando el control de la aplicación al usuario y permitiéndole interactuar con los elementos contenidos en la pantalla?	**2.5**	**2.5**	**12.5**
Correspondencia entre el sistema y el mundo real	¿La interfaz muestra mensajes en el idioma del usuario, cuando se habla de idioma se refiere a palabras, frases y conceptos familiares para el usuario, siempre en el contexto de la aplicación?	**12.5**	**10**	**12.5**
Reconocer antes que recordar	¿El diseño de la interfaz permite reducir la carga de memoria para un usuario final, se refiere a que si la interfaz ayuda al usuario a no tener que recordar información para ir de un proceso a otro a la hora de realizar una iteración?	**2.5**	**2.5**	**7.5**
Prevención de errores	¿La Aplicación tiene realizar un buen diseño de mensajes de error que den la posibilidad al usuario de retraerse antes de que se realice la acción y se comprometan los datos?	**12.5**	**12.5**	**12.5**
Estética y diseño minimalista	¿Los mensajes de la aplicación contienen información relevante para la tarea que está realizando el usuario, por otro lado el diseño de la interfaz es simple, fácil de aprender, fácil de usar y con fácil acceso a las funcionalidades que ofrece la aplicación?	**7.5**	**7.5**	**12.5**
Ayudar a los usuarios a reconocer, diagnosticar y recuperar errores	¿La aplicación tiene mensajes de error en lenguaje entendible por el usuario y sin código de lenguajes de programación, los mensajes indican el error y sugieren como solucionarlo?	**12.5**	**10**	**10**
Ayuda y documentación	¿La aplicación tiene manual de usuario, la información es fácil de encontrar y enfocada a la tarea que el usuario realiza, se listan los pasos necesarios para la realización de la tarea?	**12.5**	**12.5**	**12.5**
Total		**70**	**65**	**92.5**

Tabla 6 : resultados de la evaluación realizada en la Tabla 5.

2.11 GRÁFICA COMPARATIVA DE LA Tabla 6.

Para realizar la gráfica se denominan a las ocho heurísticas de la siguiente forma:

- ✓ **H1**: Visibilidad del estado del sistema.
- ✓ **H2:** Control y libertad del usuario.
- ✓ **H3:** Correspondencia entre el sistema y el mundo real.
- ✓ **H4:** Reconocer antes que recordar.
- ✓ **H5:** Prevención de errores.
- ✓ **H6:** Estética y diseño minimalista.
- ✓ **H7:** Ayudar a los usuarios a reconocer, diagnosticar y recuperar errores.
- ✓ **H8:** Ayuda y documentación.

La Ilustración 26, expresa de manera gráfica las distancias entre los valores reales. Permite evaluar el desempeño de una herramienta respecto a las 8 heurísticas definidas.

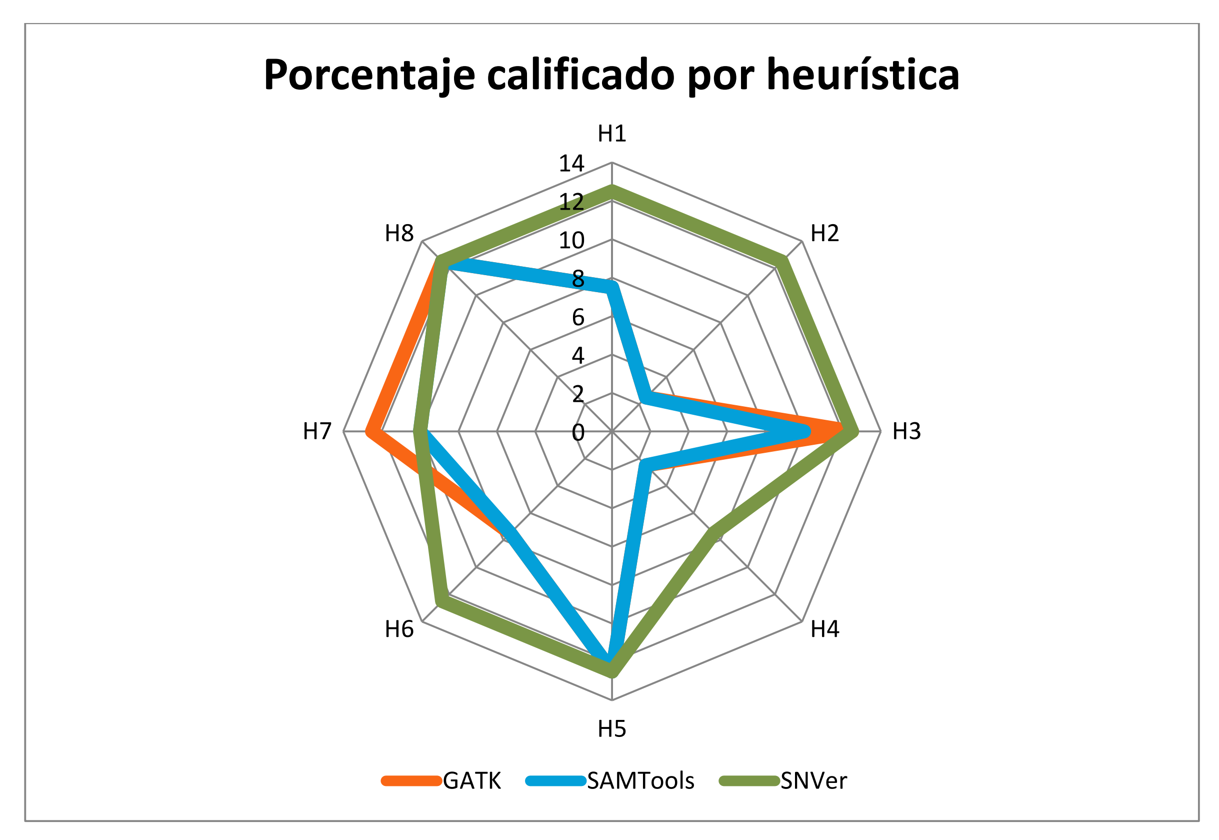


Ilustración 26: Grafica producto de los valores obtenidos por cada una de las herramientas evaluadas respecto a las 8 heurísticas de usabilidad.

De acuerdo a los resultados obtenidos se puede concluir que SNVer es muy superior respecto a la heurística uno (Visibilidad del estado del sistema) al contrario que GATK y SAMTools, en este criterio tan relevante para una aplicación como lo es mantener siempre informado al usuario del estado del sistema. En otro aspecto, importante resaltar la excelente calificación obtenida por las tres herramientas en la heurística 5 (Prevención de errores), de igual forma las tres herramientas obtienen igual resultado para la heurística 8 (Ayuda y documentación). Conforme estas calificaciones se pueden obtener el porcentaje total de usabilidad que tienen las herramientas SNVer, GATK y SAMTools. En este sentido, la Ilustración 27 representa dichos porcentajes.

2.12 GRÁFICA TOTAL DE USABILIDAD

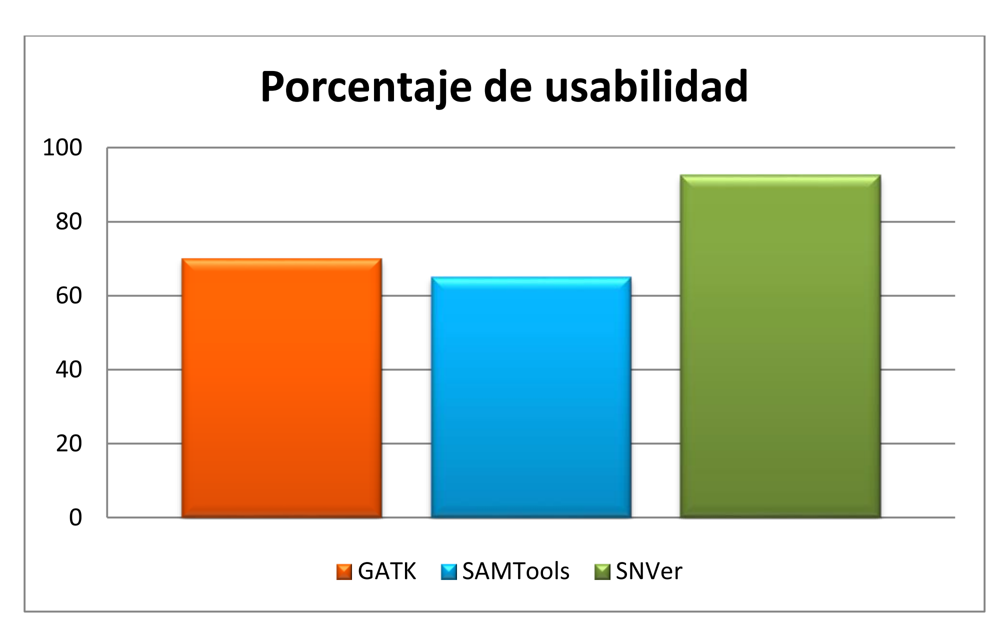


Ilustración 27: Grafica producto del porcentaje total obtenido por cada una de las herramientas evaluadas respecto a las 8 heurísticas de usabilidad.

Acorde a los resultados presentados en la Ilustración 27, se concluye que SNVer presenta mayor usabilidad puesto que es la única de las tres herramientas que implementa una interfaz gráfica de usuario (GUI), no obstante GATK y SAMtools presentan una usabilidad buena sin tener que hacer uso de una interfaz gráfica, sin embargo la implementación de una interfaz gráfica de usuario permite subir en gran porcentaje la usabilidad de la herramienta cualquiera que la implemente.

CAPÍTULO 3: DESARROLLO DE NGSEP

A continuación se presenta la Especificación de Requisitos de Software (ERS) de NGSEP (NGSTools Eclipse Plug-in) con el objetivo de definir las funcionalidades y fronteras del sistema a construir. Se definen usuarios y su papel dentro de la herramienta, listado de requerimientos funcionales y no funcionales, además se proporciona información del desarrollo de NGSEP dentro de Eclipse IDE.

3.1 FRONTERAS DEL SISTEMA

- ✓ El sistema garantiza su uso en plataformas como Windows, Ubuntu y Mac.
- ✓ El sistema se encargará de encontrar variantes genómicas.
- ✓ El sistema se encargará de acoplarse a la arquitectura de Eclipse Plug-in.

3.2 ACTORES DEL SISTEMA

El sistema consta de un solo tipo de usuario ya que se ejecutará en la máquina del usuario final de forma independiente a otras máquinas o sistemas. El usuario se denominara:

- ✓ *Biólogo*

A continuación se explica el papel del usuario dentro de NGSEP:

Biólogo: su rol dentro de NGSEP es de usuario final, el cual tendrá a su disposición todas las funcionalidades brindadas para descubrimiento de variantes genómicas.

3.3 REQUERIMIENTOS FUNCIONALES

Un requerimiento funcional define una función del sistema de software o sus componentes. Una función es descrita como un conjunto de entradas, comportamientos y salidas [42].

3.4 LISTA DE REQUERIMIENTOS FUNCIONALES

El sistema debe de permitir:

1. La comparación entre un genoma de referencia y lecturas genómicas esto, con el fin de poder realizar re secuenciación.

2. Comparar un archivo BAM, con las lecturas de un genoma y un genoma de referencia, con el fin de encontrar variabilidad genética.

3. Llevar a cabo el ordenamiento de un archivo BAM.

4. El emparejamiento de pares de lecturas que encajan en una posición de un mismo fragmento secuenciado.

5. Comparar un catálogo de variantes, un catálogo de anotaciones de genes y un genoma referencia, con el objetivo de buscar posibles variaciones o cambios con respecto al genoma de referencia y como pueden influir en la función de los genes.

6. Llevar a cabo la comparación de lecturas de un genoma (archivo BAM) contra su referencia (archivo Fasta) con el fin de organizar las lecturas alineadas con respecto a la referencia, generando un archivo de salida con los alineamientos únicos y simples de cada lectura.

7. Mezclar tres archivos con variantes y comparar contra la referencia en búsqueda de las posiciones que se encuentran con variación.

8. Determinar la cantidad de lecturas que cubre cada posición del genoma.

3.5 LISTA DE REQUERIMIENTOS NO FUNCIONALES

El sistema debe de permitir:

1. Tener procesos Sincrónicos.
2. Ser multiplataforma.
3. Tener un registro de las actividades de los procesos comprendidos dentro de NGSTools.
4. Integrarse a Eclipse IDE.
5. Debe tener interfaz grafica
6. Integrar la interfaz gráfica a NGSTools.
7. Recordar últimos archivos utilizados en los diferentes procesos.
8. Crear Jobs que monitoreen los diferentes procesos de comienzo a fin de la ejecución de los mismos.
9. Crear historial de los archivos utilizados en el proceso de detección de variantes de NGSTools.
10. Cargar rutas de los archivos que se van a utilizar dentro del aplicativo.
11. Generar archivos con los datos generados en cada proceso.
12. Generar graficas de cobertura.
13. Generar graficas de calidad.
14. Correr el proceso de mapeo si está instalado Bowtie2 en el ordenador.

3.6 ECLIPSE IDE

Como es definida en la página Web oficial (www.eclipse.org): la Plataforma Eclipse es un IDE para todo y nada en particular), una poderosa herramienta que permite integrar diferentes aplicaciones para construir entornos de desarrollo integrado (IDEs) que pueden ser utilizados para la construcción de aplicaciones Web, JavaTM, C/C+, entre otras, dando a los desarrolladores la libertad de elegir en un entorno multilenguaje y multiplataforma. Es un proyecto de desarrollo de software open- source, que está dividido en tres partes:

- ✓ The Eclipse Project: es un proyecto de desarrollo de software libre destinado a proporcionar una plataforma de desarrollo de herramientas integradas robusta, completa y comercial. Se subdivide, a su vez, en tres subproyectos:

- ✓ La propia plataforma, que contiene las herramientas Eclipse.

- ✓ JDT (Java Development Toolkit): añade a la plataforma un IDE de Java completamente equipado, incluyendo: editor, refactoría (permite preservar la semántica de un programa), compilador y depurador.

- ✓ PDE (Plug-in Development Environment): es un conjunto de herramientas diseñadas para ayudar al desarrollador de Eclipse en las tareas de desarrollo, prueba, depuración, construcción y distribución de Plug-ins [26].

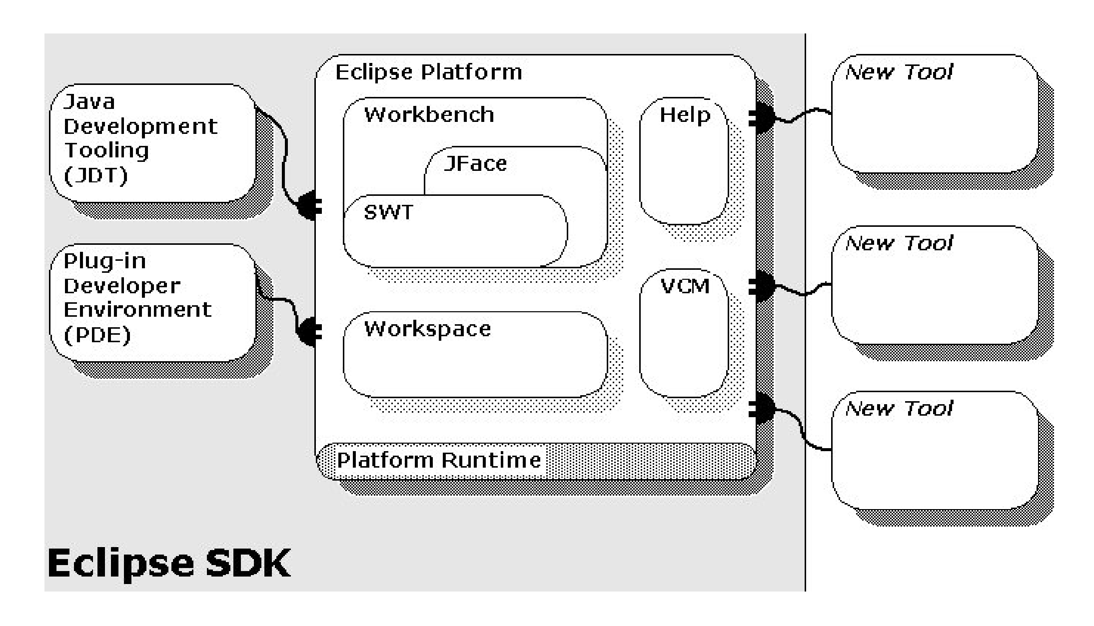


Ilustración 28: Entorno de trabajo de Eclipse [27].

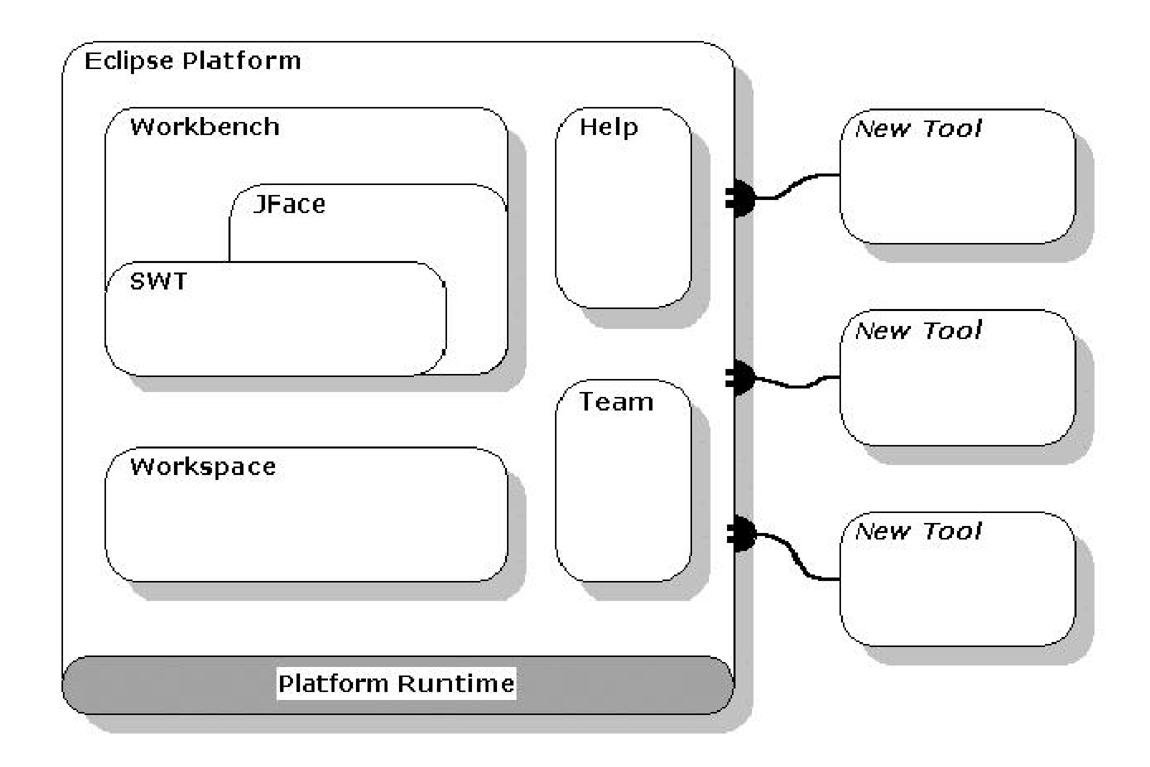


Ilustración 29: Arquitectura de la Plataforma Eclipse [27].

3.7 ARQUITECTURA DE LA PLATAFORMA ECLIPSE

Considerándola desde términos de diseño, la plataforma eclipse no ofrece gran funcionalidad por sí sola, si no que su valor real yace en el modelo de Plug-ins con lo cual, eclipse está estructurada como un conjunto de subsistemas los cuales son implementados en uno o más Plug-ins que corren sobre la plataforma de ejecución (Ilustración 28: Entorno de trabajo de Eclipse [27].Ilustración 28) dichos subsistemas definen puntos de extensión para permitir agregar funcionalidad a la plataforma [27].

3.8 PLATAFORMA DE EJECUCIÓN (PLATFORM RUNTIME)

"Se trata del único componente de eclipse que no es un Plug-in. Al iniciar la plataforma de ejecución se descubren de manera dinámica el conjunto de Plug-ins disponibles. Se leen sus archivos de manifiesto, y se construye en memoria un registro de Plug-ins, que está disponible a través de la API de la plataforma.

La plataforma mantiene un registro de aquellos Plug-ins instalados así como de las funcionalidades que proveen, no podrán ser añadidos nuevos Plug-ins después del inicio.

Para agregar nuevas funciones al sistema se usa un modelo de extensión común. Los "puntos de extensión" son lugares bien definidos dentro del sistema que permiten ser extendidos por Plug-ins cuando una herramienta contribuye con una implementación para

determinado punto de extensión se dice que agrega una “extensión” a la plataforma. A su vez cada Plug-in puede definir sus propios puntos de extensión de tal forma que puedan ser extendidos por otros. Este mecanismo de extensión es la única manera de agregar funcionalidad a la plataforma.

Un objetivo muy importante del runtime es que usuarios finales no sufran desventajas a causa del uso de memoria por aquellos Plug-ins, que si bien están instalados, no están siendo usados. De esta manera un Plug-in puedo ser instalado y agregado al registro pero el mismo no será activado a menos que se requiera mediante la actividad de usuario” [27].

3.9 WORKSPACE

“Se trata del bloque central, o espacio de trabajo, para los archivos regulares que son específicos de cada usuario, y sobre los que actúan las diferentes herramientas instaladas en la plataforma.

El espacio de trabajo del usuario consta de uno o más proyectos donde cada uno se mapea a un directorio especificado por el usuario en el sistema de archivos. Cada proyecto contiene los archivos que son creados y manipulados por el usuario. Todos los archivos en el espacio de trabajo son directamente accesibles por programas estándar y herramientas del sistema operativo.

El conjunto de proyectos, archivos y carpetas que son generados por herramientas y almacenados en el sistema de archivos, constituye los recursos del workspace. Están organizados en una estructura de árbol, con los proyectos arriba y los archivos y carpetas abajo” [27].

3.10 WORKBENCH

“Implementa el aspecto visual que permite al usuario navegar por otros recursos y utilizar las herramientas integradas. El workbench es simplemente un frame donde se presentan varias partes visuales, estas partes se pueden dividir en dos categorías mayores: editores y vistas” [27].

Ante la necesidad de integrar un flujo de trabajo para el análisis de datos bioinformáticos con el uso de interfaces fáciles de manejar y que estén disponibles para la comunidad científica. Se hizo indispensable el uso del workbench de eclipse como la herramienta de visualización y manipulación de los archivos que se utilizan para cada uno de los procesos de NGSTools y para los cuales se desarrolló una serie de comandos, encapsulado estos comandos bajo interfaces intuitivas y muy fáciles de usar para un usuario final.

El uso de workbench no solo aporto el ingreso a la aplicación por medio de interfaces si no que permitió visualizar y controlar el proceso de ejecución de todos los procesos de NGSTools, mediante la implementación y manipulación de la vista contenida en eclipse

denominada progress y en la cual se aplicó el patrón de diseño observador observado, logrando ver el estado en tiempo real de la ejecución, avance de dichos procesos y su posterior fin. Por medio de esta implementación también se logró detener si así deseara el usuario el avance de los procesos.

Workbench facilito adicionar en forma de menú y submenús todos los procesos que contiene NGSTools en una estructura organizada y entendible para el usuario que va hacer uso del pipeline. Permitiendo nombrar cada proceso y encapsular en vistas con entradas de muy fácil entendimiento para un usuario con pocos conocimientos o nulos de programación.

El uso de workbench fue de gran ayuda para la resolución de la problemática de la poca facilidad de las herramientas actuales y no solo contribuyo a esta solución, da la garantía o la posibilidad de hacerse extensible mediante el Plug-in (NGSEP) a las ideas que a futuro se planten con respecto a visualización y manipulación de archivos producidos por NGS.

3.11 STANDARD WIDGET TOOLKIT (SWT)

SWT (siglas en inglés de *Standard Widget Toolkit*) es un conjunto de componentes para construir interfaces gráficas en Java, (widgets) desarrollados por el proyecto Eclipse.

Recupera la idea original de la biblioteca AWT de utilizar componentes nativos, con lo que adopta un estilo más consistente en todas las plataformas, pero evita caer en las limitaciones de ésta.

La biblioteca Swing, por otro lado, está codificada enteramente en Java y frecuentemente se le acusa de no brindar una experiencia idéntica a la de una aplicación nativa. Sin embargo, el precio a pagar por esa mejora es la dependencia (a nivel de aspecto visual y no de interfaz de programación) de la aplicación resultante del sistema operativo sobre el cual se ejecuta. La interfaz del workbench de eclipse también depende de una capa intermedia de interfaz gráfica de usuario (GUI) llamada JFace que simplifica la construcción de aplicaciones basadas en SWT [28].

Por estos motivos se elegido como biblioteca grafica para NGSEP a SWT, siguiendo la exigencia del mercado de herramientas multiplataforma, porque garantiza su excelente integración al sistema operativo nativo.

Además esta biblioteca ofrece dos características únicas: es nativa y portable a la vez. Dota a las aplicaciones escritas con ella de una mayor velocidad en su ejecución y un menor consumo de recursos respecto a las aplicaciones escritas con Swing o AWT.

3.12 JFACE

Es un conjunto de widgets para realizar interfaces de usuario construido sobre SWT. Fue desarrollado por IBM para facilitar la construcción del entorno de desarrollo Eclipse, pero su uso no está limitado a éste.

JFace proporciona una serie de construcciones muy frecuentes a la hora de desarrollar interfaces gráficas de usuario, tales como cuadros de diálogo, evitando al programador la tediosa tarea de lidiar manualmente con los widgets de SWT [29].

3.13 PLUG-IN

Es un módulo de hardware o software que añade una característica o un servicio específico a un sistema más grande.

Un Plug-in es la unidad mínima de funcionalidad de Eclipse ™ que puede ser distribuida de manera separada. Herramientas pequeñas se escriben como un único Plug-in, mientas que en las complejas la funcionalidad está en varios Plug-ins.

Para añadir un Plug-in a la plataforma de Eclipse ™ existe un único modo, los puntos de extensión. En conformidad con el paradigma orientado a objetos, un punto de extensión no deja de ser una interfaz que puede ser implementada por algún desarrollador dispuesto a extender la plataforma [27].

CAPÍTULO 4: EVALUACIÓN DE NGSEP EN UN ESTUDIO DE CASO REAL

En este capítulo, conforme al contexto explicado en el capítulo uno, dos y tres, el lector encontrará un estudio de caso real en el que se ha puesto a prueba la herramienta NGSEP con secuencias del organismo levadura. Al finalizar este capítulo, se realizó una comparativa de usabilidad de (GUI) entre la herramienta NGSEP descrita en este capítulo cuatro y la herramienta SNVer ganadora de la comparación realizada en el capítulo dos pág.37.

4.1 INTRODUCCIÓN A NGSEP

El primer paso para comenzar a utilizar NGSEP es definir los archivos con los que se va trabajar el pipeline ofrecido por la herramienta, en este sentido la prueba que se mostrará a continuación se va realizar con secuencias de levadura.

Definidos los archivos con los que se va trabajar se debe instalar NGSEP en Eclipse en caso de que no esté instalado, para esta prueba se supone que ya está instalado NGSEP en Eclipse.

Se utiliza como entorno de trabajo el sistema operativo Windows.

- ✓ El Eclipse que se va utilizar es Juno 4.2.2

Lista de pasos:

1. Abrir Eclipse Juno 4.2.2, después de tener instalado el Plug-in NGSEP.
2. Crear un proyecto, en este caso se va crear un simple Project porque vamos a manipular archivos, no hacer programación o cualquier otro tipo de proyecto de desarrollo en Eclipse.
3. El nombre del proyecto será “prueba Levadura”.
4. Luego de crear el proyecto se procede a ingresar los archivos con los que se realizó esta prueba.

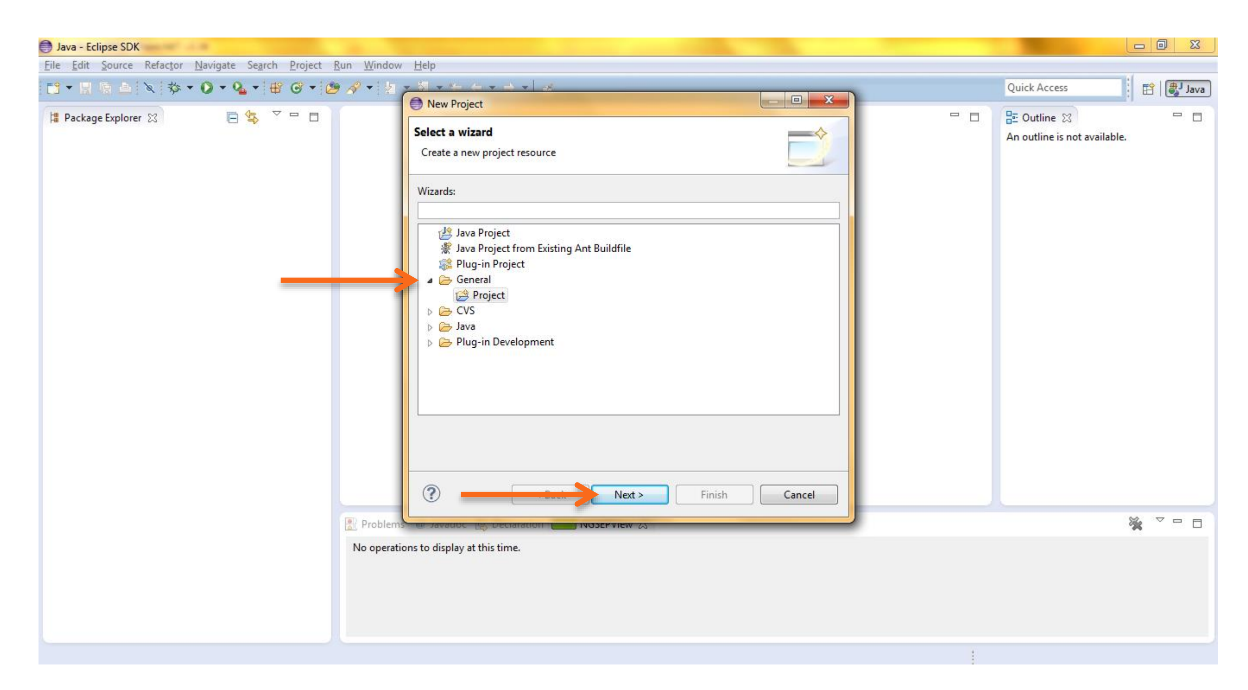


Ilustración 30: creando un general Project de Eclipse para empezar a trabajar con NGSEP.

4.2 ARCHIVOS GENÉTICOS

En este caso de estudio, se optó por trabajar con un par de lecturas que son complementarias de una secuencia de levadura. A partir de este par de lecturas se generan diferentes archivos de forma secuencial en cada uno de los diferentes procesos ofrecidos por NGSEP.

✓ Nombre de las lecturas: Samplen47_Cleandata_1.fq, Samplen47_Cleandata_2.fq.

Como se puede ver en la Ilustración 31, están en formato FASTAQ, el cual es explicado en las primeras páginas de este documento.

Para tener en cuenta, es indispensable tener el genoma de referencia para lecturas a las cuales se deseé detectar variantes genómicas, en este caso el genoma de referencia de levadura es:

✓ Nombre genoma de referencia: sacCer_SGD_refgenome_20110301.fa.

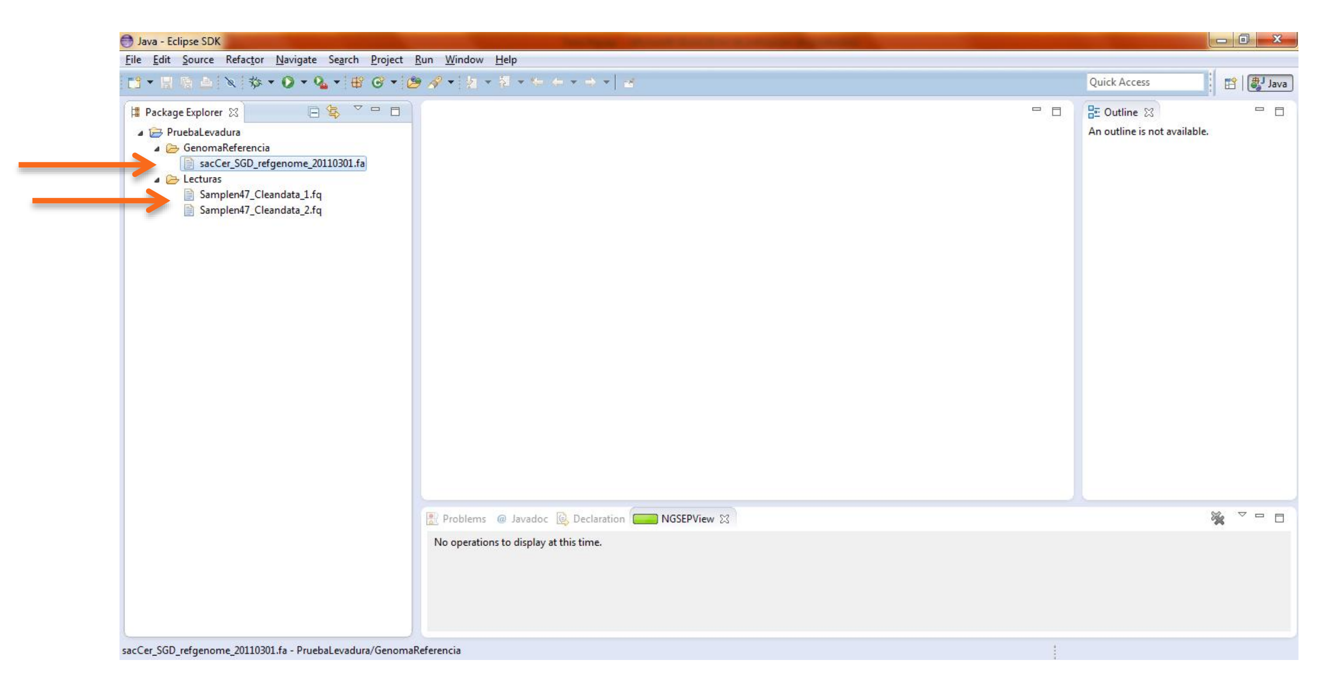


Ilustración 31: proyecto ''PruebaLevadura'' con dos lecturas de levadura y el genoma de referencia de levadura.

4.3 EJECUTANDO NGSEP

4.3.1 CREAR INDICE DE BOWTIE2

El primer paso para realizar detención de variantes genómicas en NGSEP, es indexar el genoma de referencia con el que se va trabajar. Se debe crear un índice para la referencia porque, en NGSEP el proceso de mapear lecturas lo hace Bowtie2 un programa especializado en realizar alineamientos (mapeo) de secuencias haciendo uso de datos NGS.

Según las especificaciones de Bowtie2, se debe indexar la referencia por rendimiento en términos de tiempo y funcionamiento en la memoria del sistema que haga uso de la herramienta. "Bowtie2 construye un índice a partir de un conjunto o de una sola secuencia de ADN, y con Bowtie2-build genera un conjunto de seis archivos con sufijos 0.1. BT2, 0.2. BT2, 0.3. BT2, 0.4. BT2, rev.1.bt2 y rev.2.bt2. Estos archivos juntos constituyen el índice: son todo lo que se necesita para alinear lecturas. Los archivos FASTA, que son las secuencias originales ya no son utilizados por Bowtie2 una vez que el índice se construye" [39].

Este proceso de indexación es posible realizarlo con NGSEP siempre en cuando se tenga instalado Bowtie2 en la máquina. Para acceder a indexar el genoma de referencia se debe dar clic derecho sobre el archivo que previamente se ingresó en el proyecto, a continuación se desplegara una serie de menús, dentro de estos menús se encuentra la opción NGSEP Menú, una vez localizada esta opción se procede a ubicar el puntero del mouse sobre está, se puede observar inmediatamente que se desplegara una serie de submenús al lado derecho, dentro de estos submenús organizados estratégicamente de acuerdo el pipeline

de NGSEP encontrará de primero el proceso de indexación el cual recibe el nombre de *“Create Index Bowtie”* en cual tendrá que dar clic.

En este sentido, la siguiente Ilustración 32 ayudará para comprender la serie de pasos para acceder a *“Create Index Bowtie”* Ilustración 33.

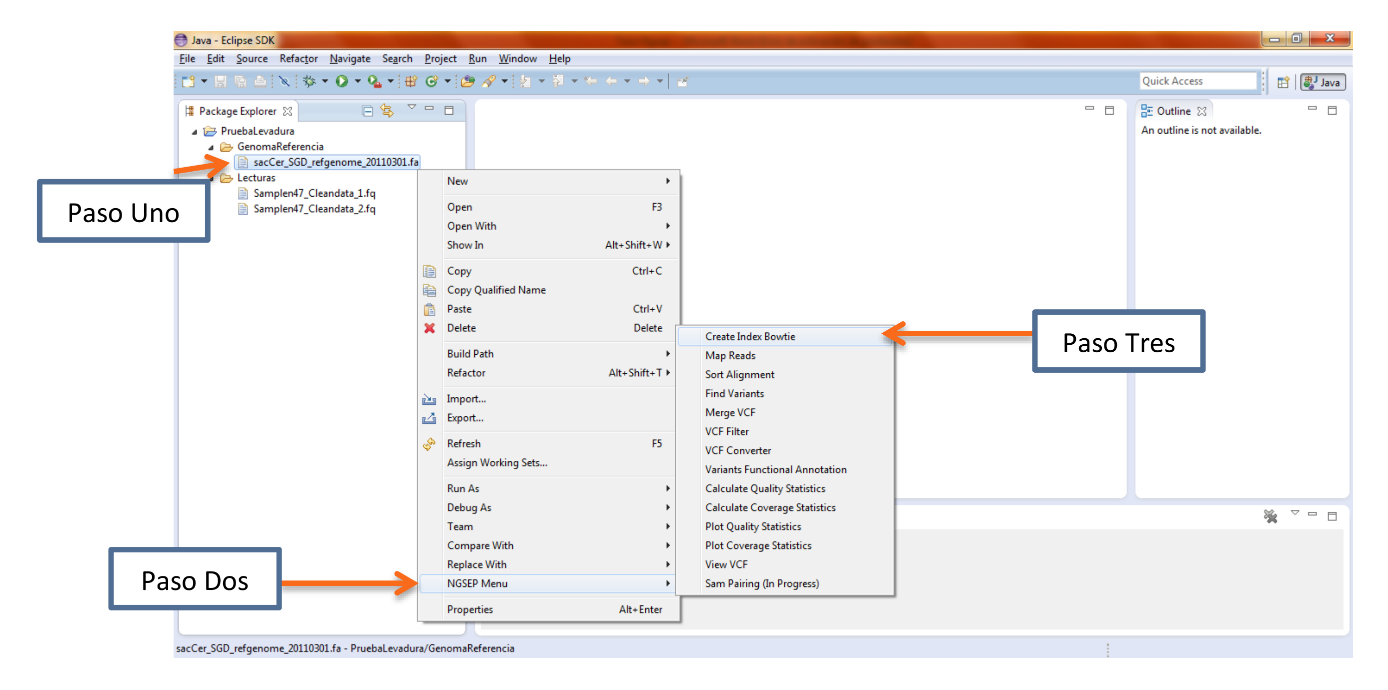


Ilustración 32: accediendo al proceso crear índice de bowtie.

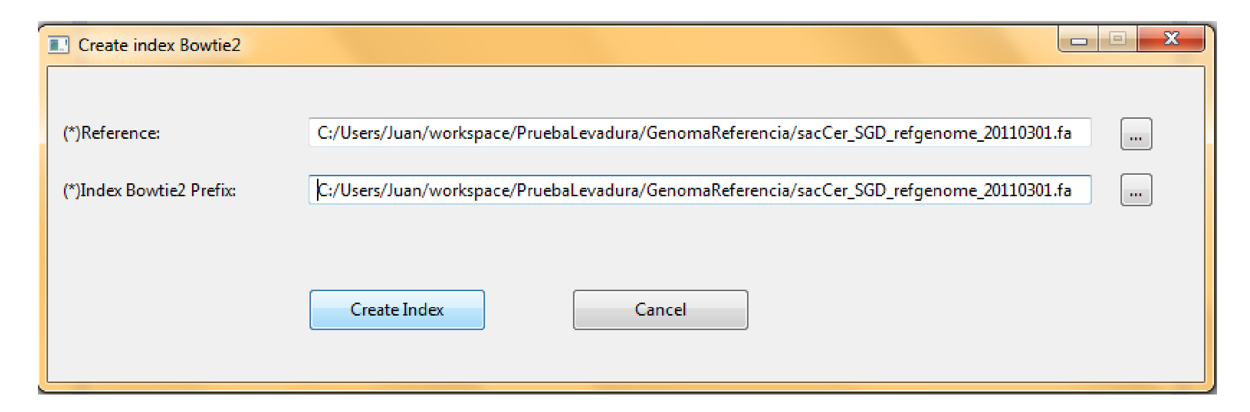


Ilustración 33: pantalla de ‘’create index bowtie’’.

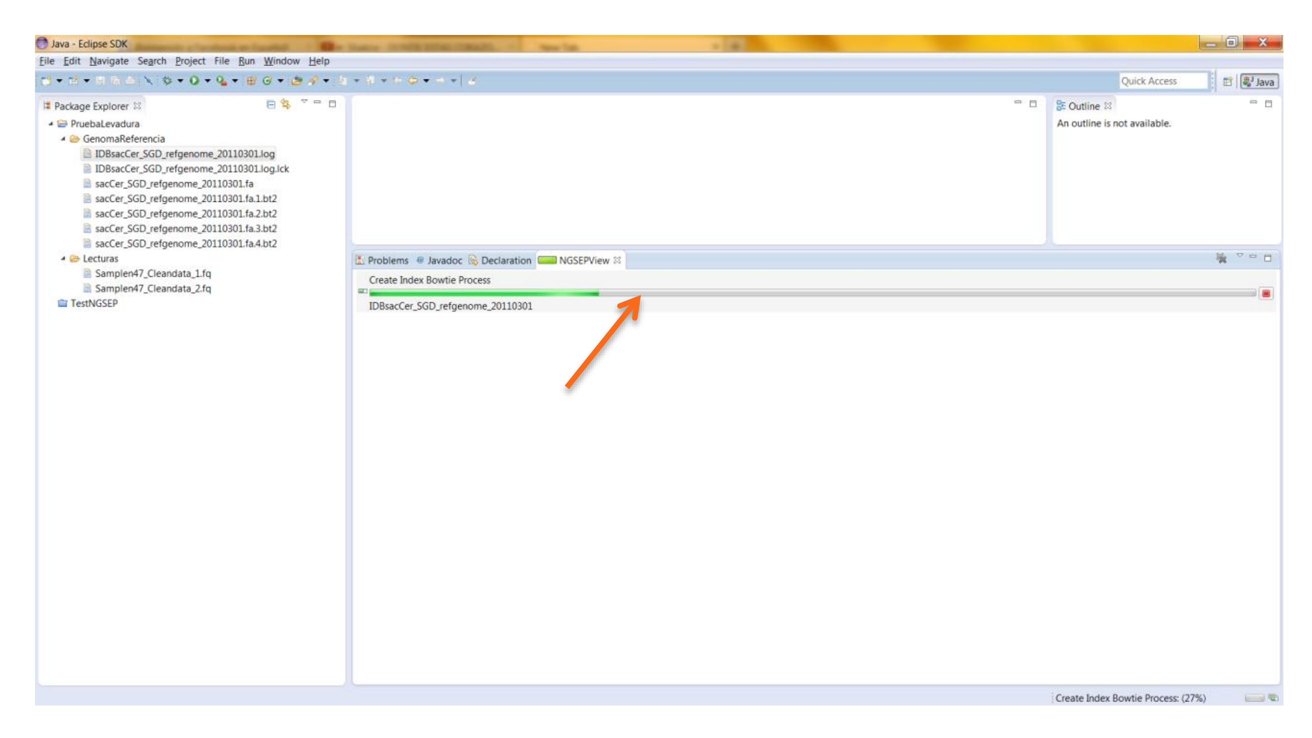


Ilustración 34: barra de progreso generada por "Create index bowtie''.

Una vez finalizado el proceso se podrá observar el índice de bowtie2 repartido en los seis c archivos generados.

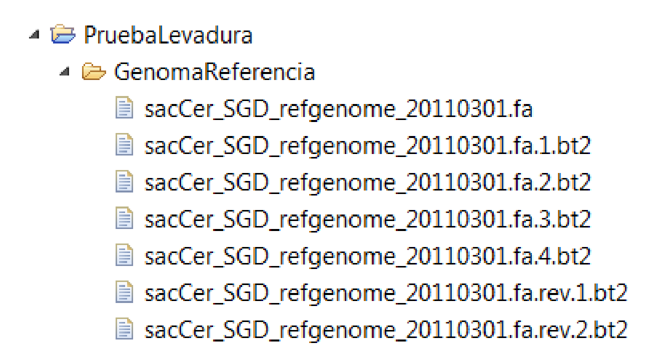


Ilustración 35: archivos generados por el proceso "Create index bowtie''.

Este proceso es vital para poder realizar el mapeo de las lecturas, sin tener la referencia indexada no es posible realizar mapeo.

4.3.2 MAPEO DE LECTURAS

Luego de terminar el proceso anterior donde se creó el índice para el genoma de levadura, se procede a continuar con el segundo paso en el pipeline de NGSEP que es realizar el mapeo de las dos lecturas de una secuencia de levadura explicadas en las páginas anteriores.
Realizar mapeo es importante porque permite alinear cada una de las lecturas en la posición adecuada utilizando el genoma de referencia como guía, esta opción se realiza con Bowtie2 que es llamado por NGSEP.

Para acceder a esa opción se selecciona las dos lecturas Samplen47_Cleandata_1.fq y Samplen47_Cleandata_2.fq, luego se da clic derecho y se accede a la opción "Map Reads" de NGSEP Menú.

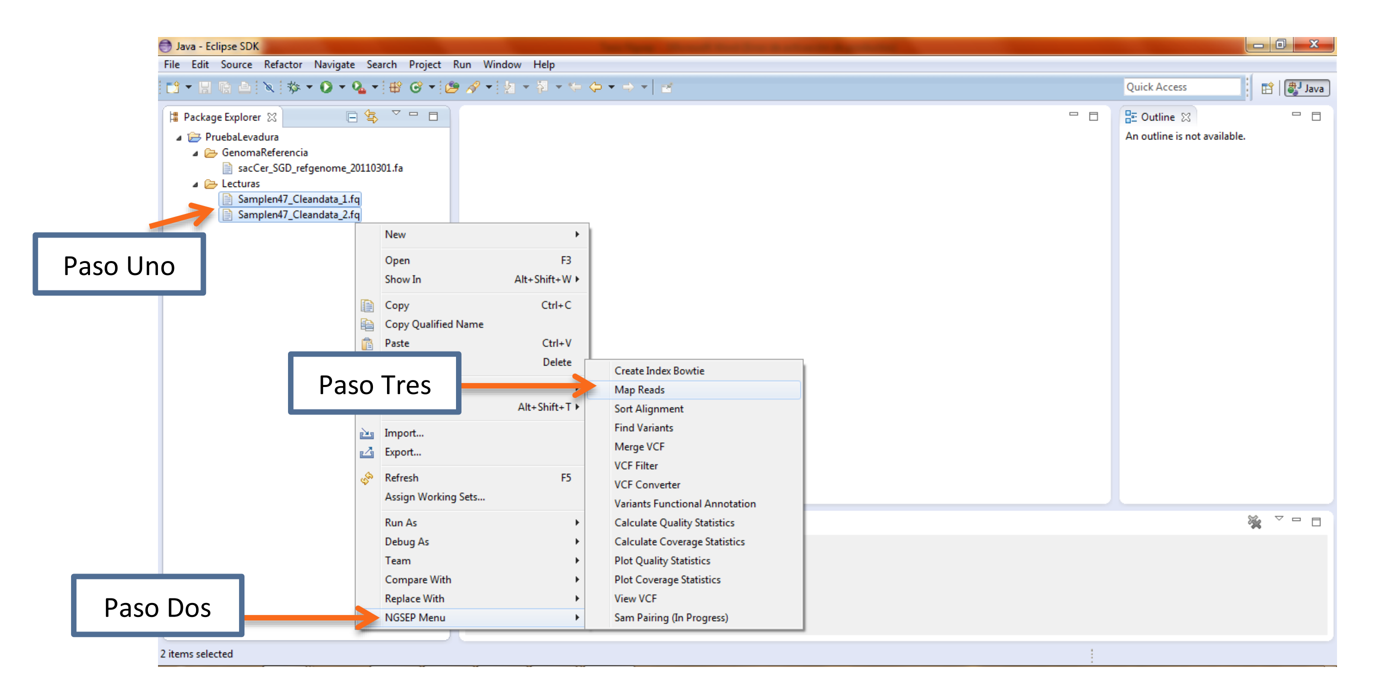


Ilustración 36: accediendo al proceso ''Map Reads''.

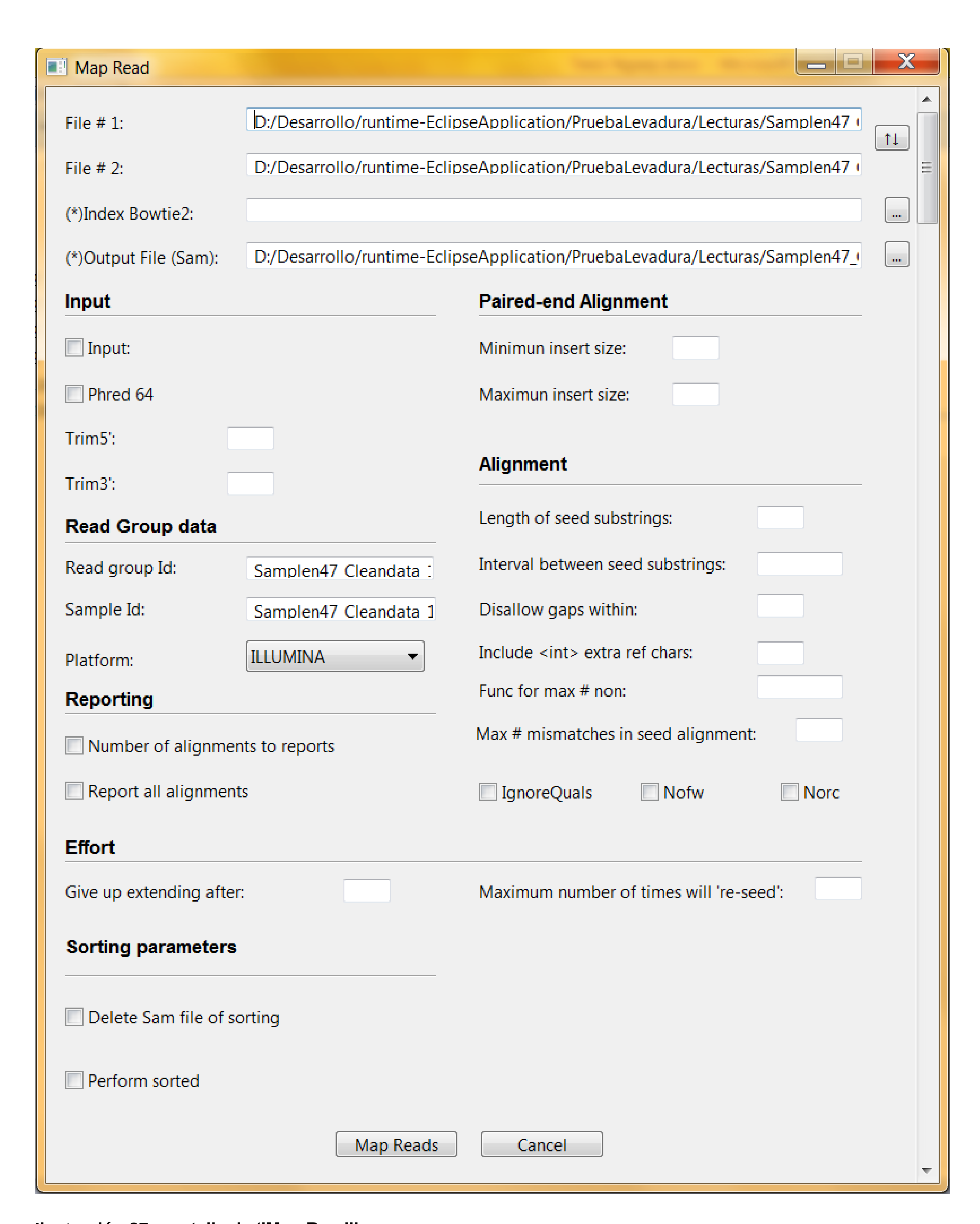


Ilustración 37: pantalla de ''Map Read''.

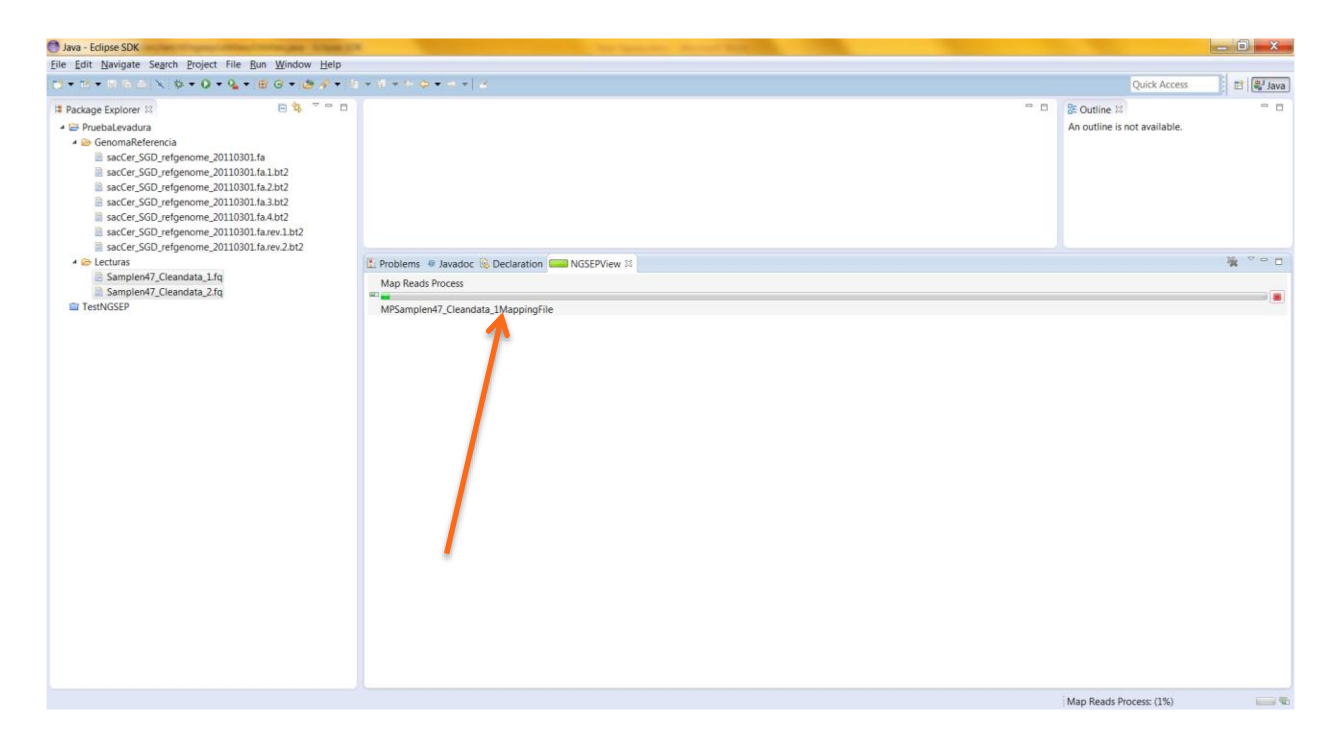


Ilustración 38: barra de progreso generada por el proceso ''Map Reads''.

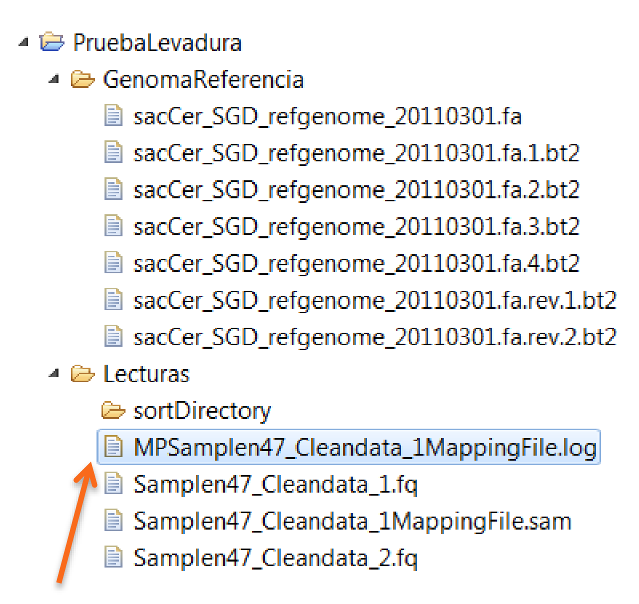


Ilustración 39: archivos generados por el proceso de ''Map Reads''

Una vez finalizado el proceso de Map Reads, se genera el archivo SAM con el resultado de alinear las lecturas en las posiciones del genoma en las que empaten.

4.3.3 ORDENAMIENTO DE ARCHIVO SAM

Luego de llevar a cabo el proceso de mapeo y tener el archivo SAM con los alineamientos se procede a comenzar el tercer proceso de NGSEP, donde se va comprimir el archivo SAM en un archivo BAM de menor tamaño y en formato entendible para la máquina.

Para acceder a la opción, se debe seleccionar el archivo SAM y dar clic derecho sobre él. Luego de esto se procede de igual forma que en los procesos anteriores a buscar NGSEP menú y luego la opción Sort Alignment.

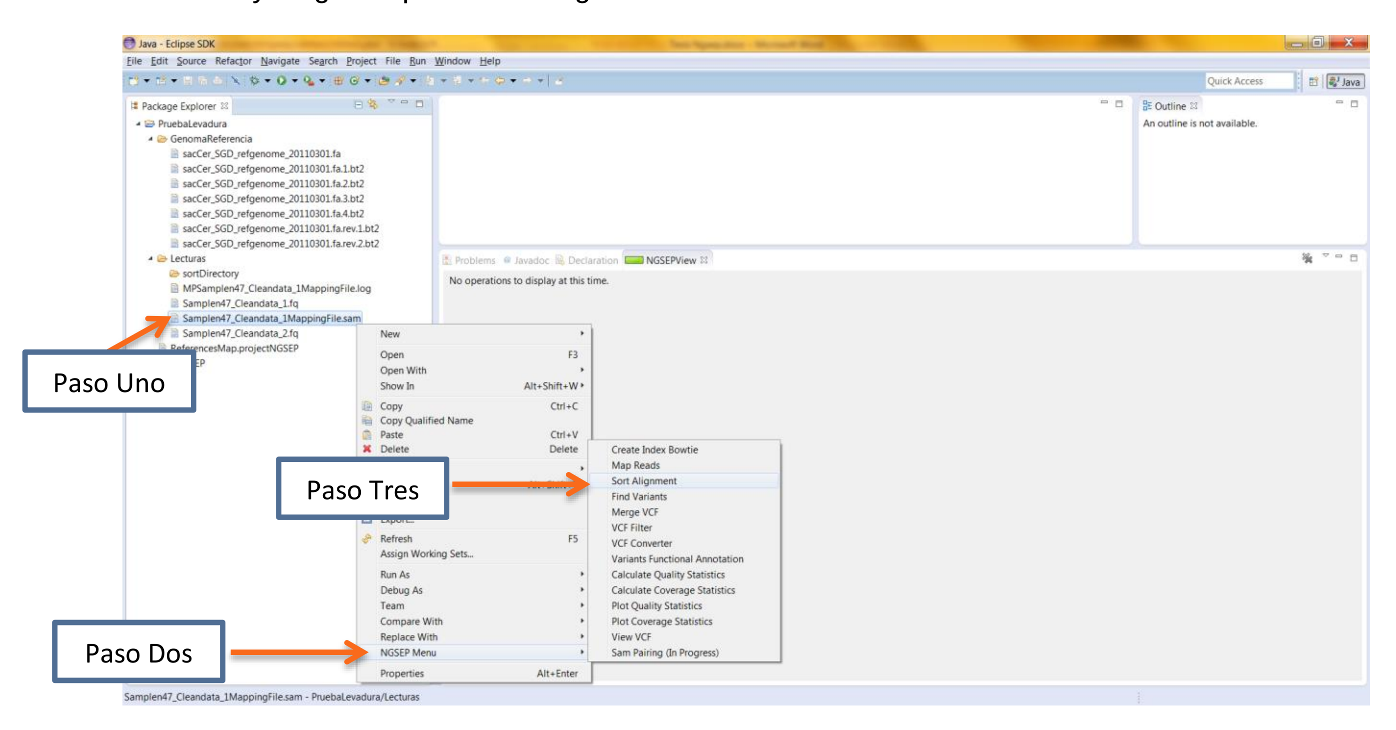


Ilustración 40: accediendo a ''Sort Alignment''.

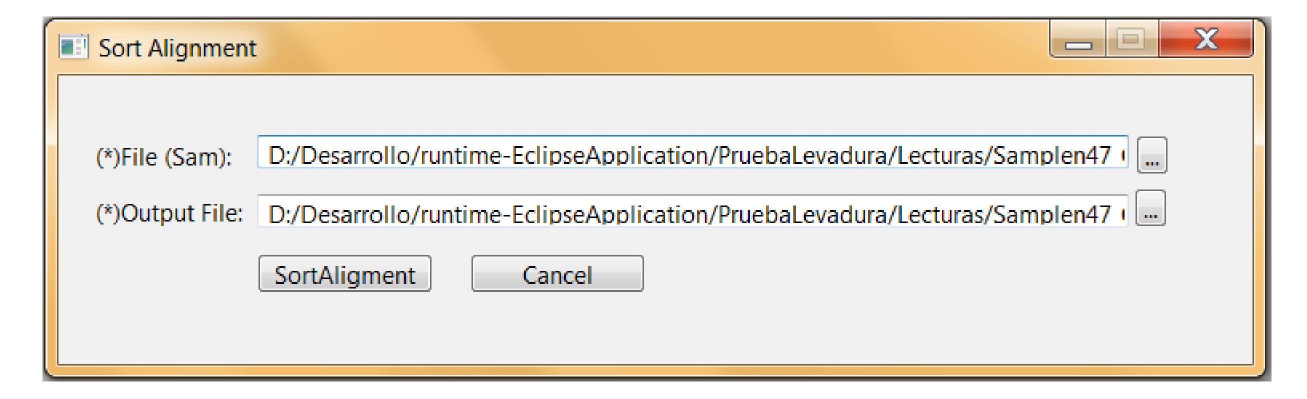


Ilustración 41: pantalla de ''Sort Alignment''.

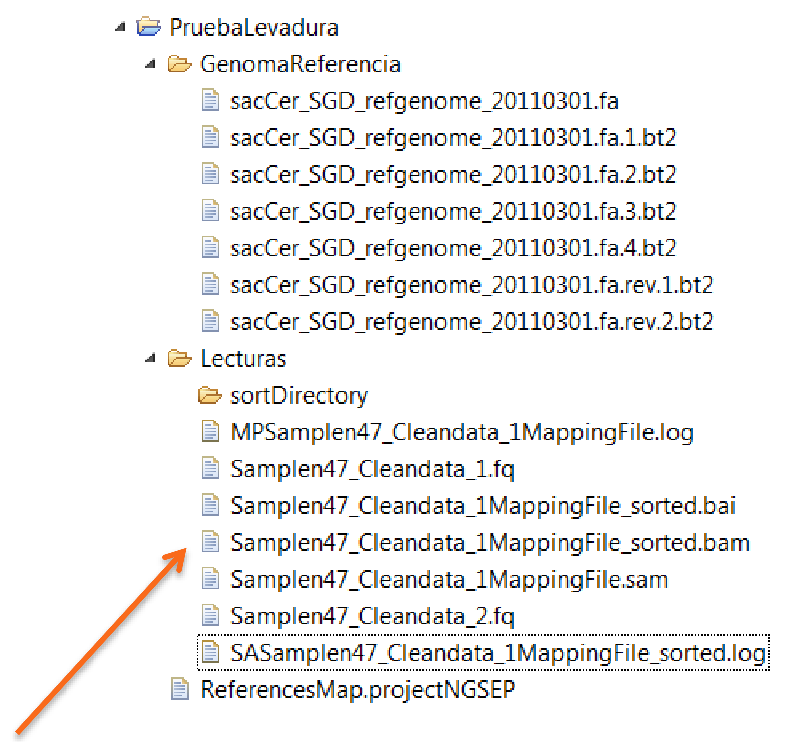


Ilustración 42: resultados arrojados por el proceso de ''Sort Alignment''.

Una vez finalizado el proceso de SORT ALIGNMENT se generan el archivo BAM con las lecturas comprimidas y un archivo bai. Este archivo bai es un índice que permite a los programas que puedan acceder y leer el archivo BAM de una manera eficiente.

4.3.4 DETENCCIÓN DE VARIANTES

Este proceso es el más importante dentro del pipeline de NGSEP, porque aquí es donde se realiza la detección de variantes genómicas producto de la comparación del archivo BAM generado en el proceso tres contra el genoma de referencia de levadura.

Para acceder a este proceso se selecciona el archivo BAM de nombre "Samplen47_Cleandata_1MappingFile_sorted.bam" y dar clic derecho sobre este, luego buscar la opción Find Variants dentro de NGSEP menú.

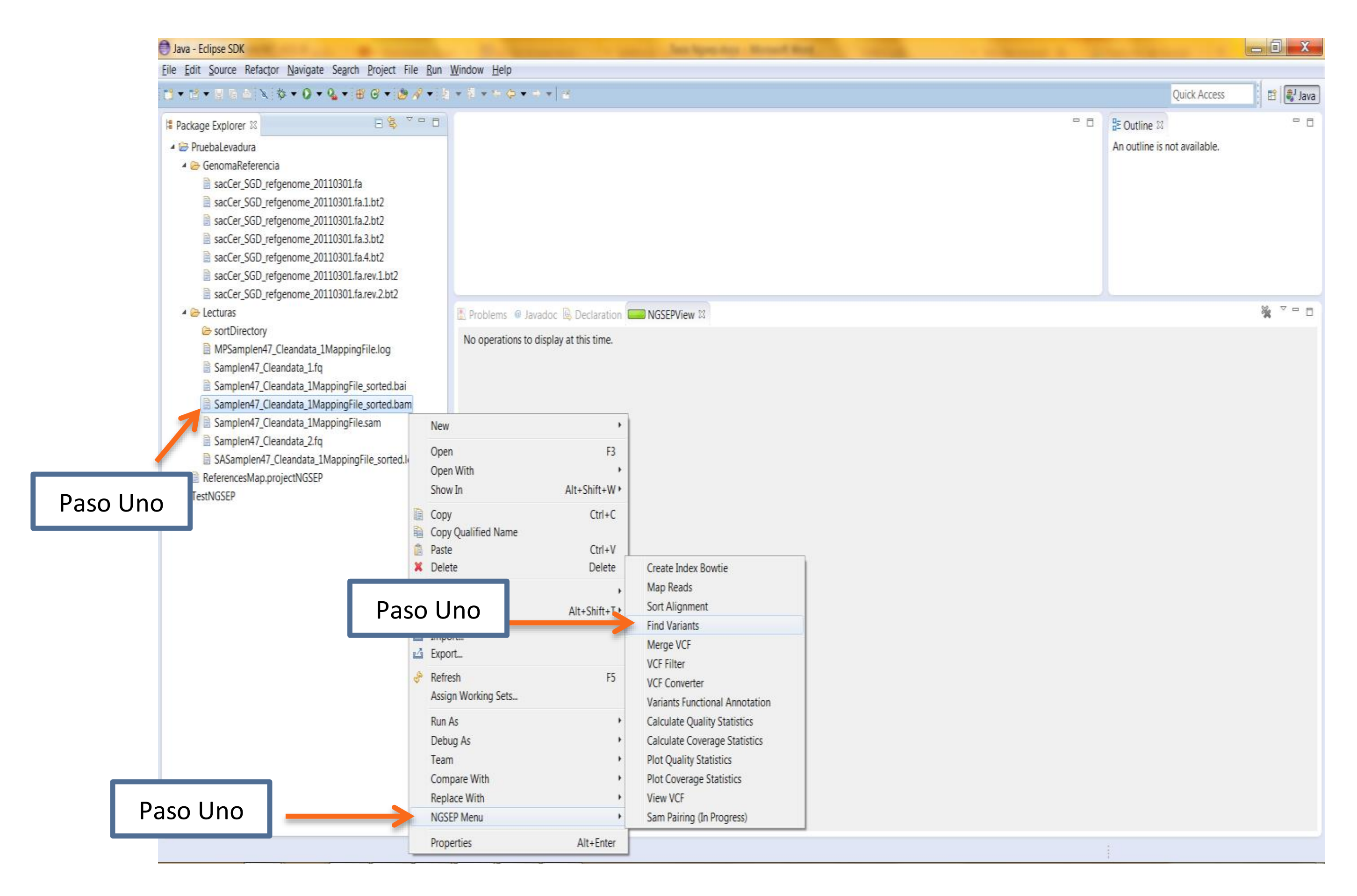


Ilustración 43: accediendo al proceso ''Find Variants''.

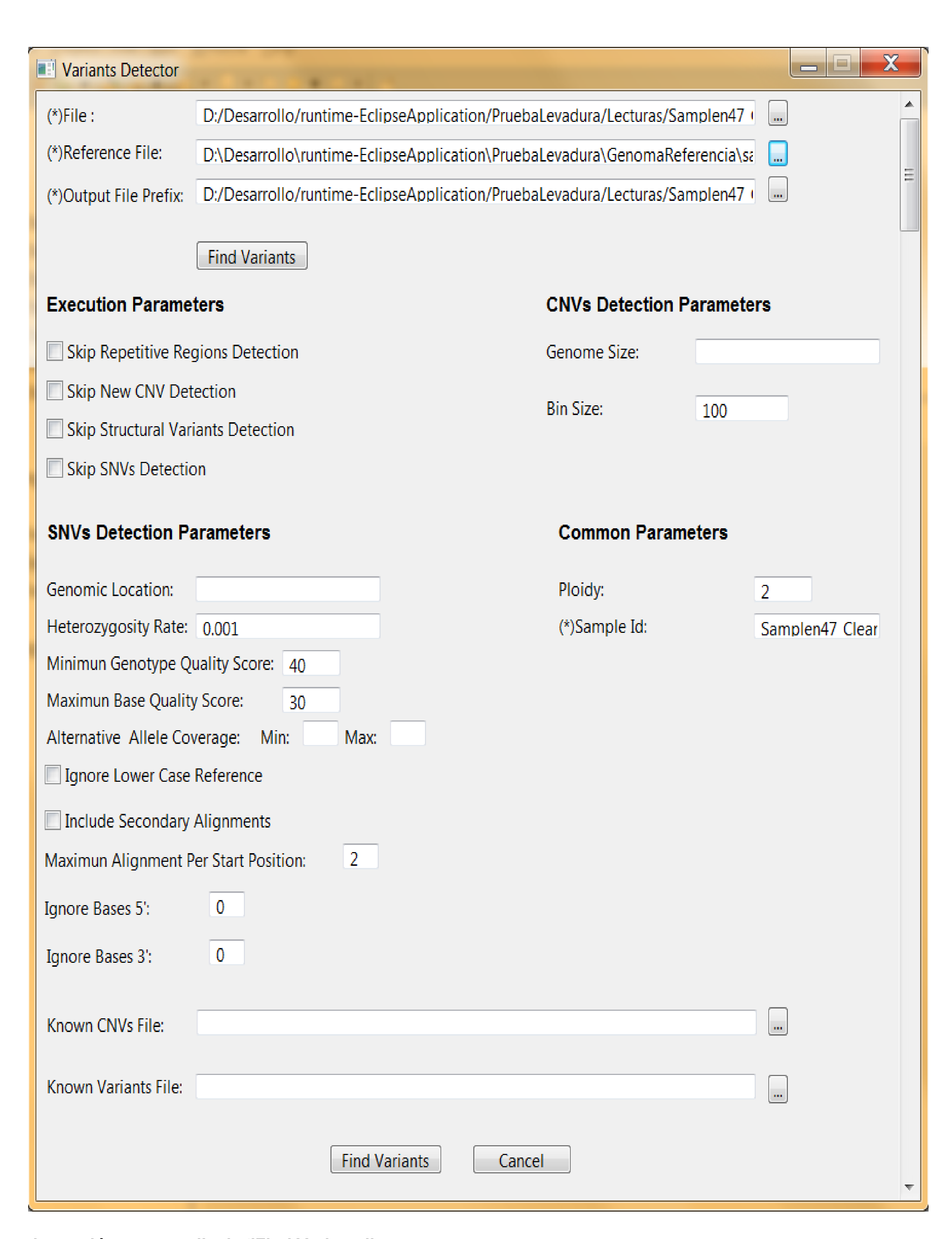


Ilustración 44: pantalla de ''Find Variants''.

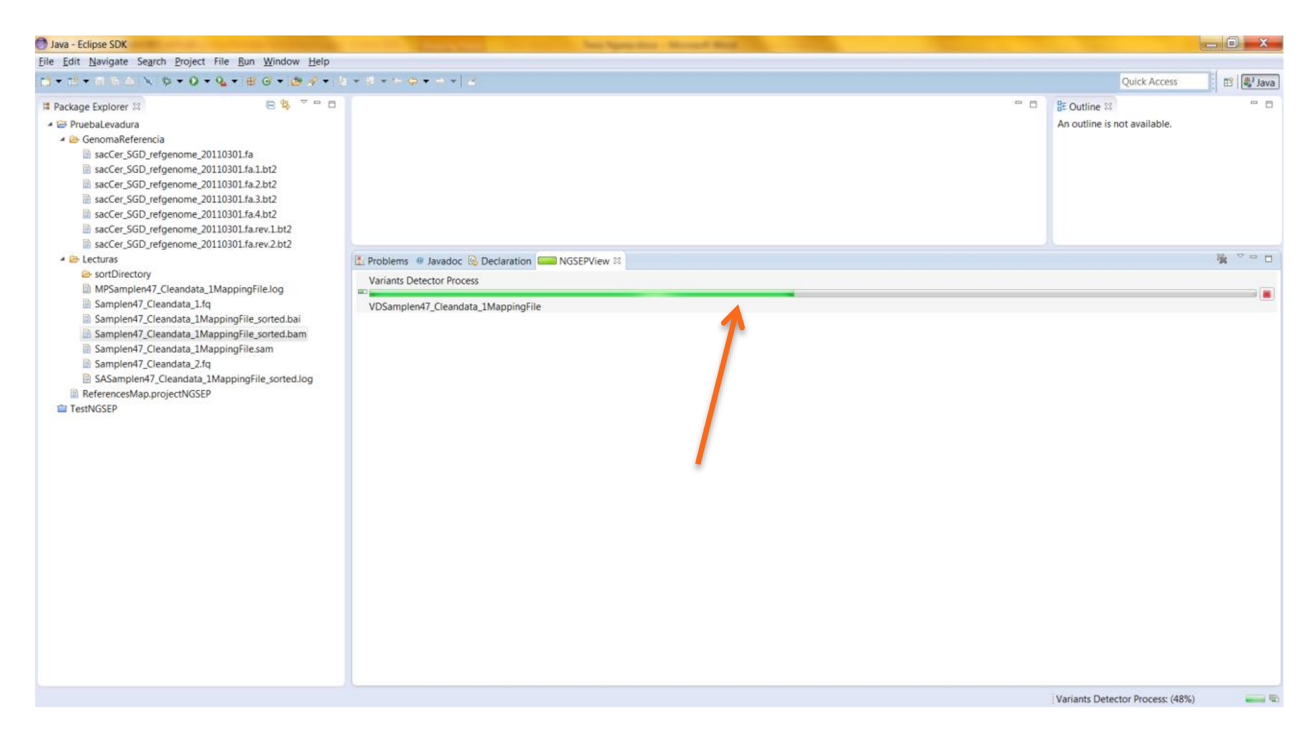


Ilustración 45: barra de progreso generada por el proceso ''Find Variants''.

Ejecución de Find Variants.

El proceso de ejecución lleva a cabo la comparación entre el genoma de levadura y la muestra o lectura Sample47 que es una extracción de la secuenciada de una planta de levadura, esta comparación va arrojar un archivo VCF con todas las variantes genómicas encontradas durante la comparación posición a posición del genoma de referencia.

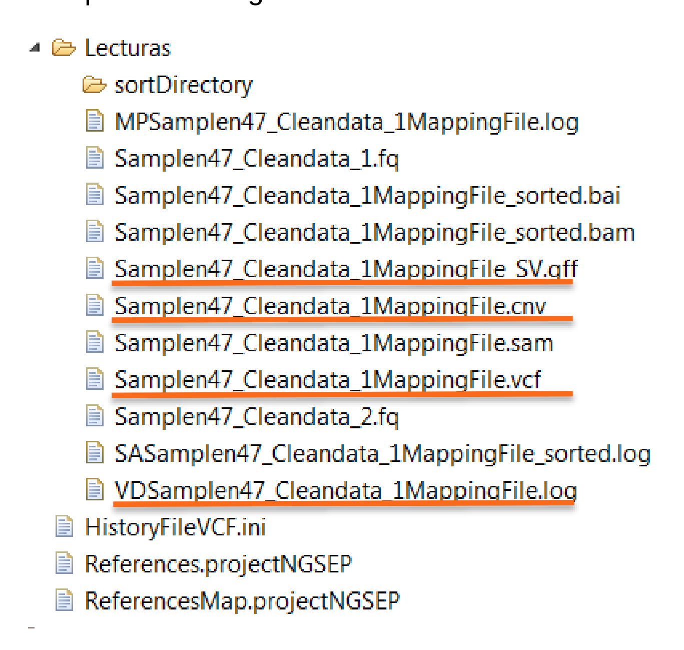


Ilustración 46: archivos generados por el proceso ''Find Variants''.

El resultado de la ejecución de Find Variants arroja los archivos "Sample47_Cleandata_1MappingFile.cnv", "Sample47_Cleandata_1MappingFile.vcf", "Sample47_Cleandata_1MappingFile_SV.cnv" y el historial de Fin Variants "HistoryFileVCF.ini".

Para verificar el resultado de la detección de variantes genómicas se debe abrir el archivo VCF, a continuación se muestra una parte del archivo.

```
Samplen47_Cleandata_1MappingFile.vcf
1  ##fileformat=VCFv4.1
2  ##INFO=<ID=CNV,Number=1,Type=Integer,Description="Number of samples with CNVs around this variant">
3  ##INFO=<ID=TA,Number=1,Type=String,Description="Variant annotation based on a gene model">
4  ##INFO=<ID=TID,Number=1,Type=String,Description="Id of the transcript related to the variant annotation">
5  ##INFO=<ID=TGN,Number=1,Type=String,Description="Name of the gene related to the variant annotation">
6  ##INFO=<ID=TCO,Number=1,Type=Float,Description="One based codon position of the start of the variant. The decimal is the codon position">
7  ##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
8  ##FORMAT=<ID=GL,Number=G,Type=Float,Description="Genotype likelihoods">
9  ##FORMAT=<ID=GP,Number=G,Type=Integer,Description="Genotype posterior probabilities">
10 ##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype quality">
11 ##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read depth">
12 ##FORMAT=<ID=AC,Number=A,Type=Integer,Description="Counts for observed alleles">
13 ##FORMAT=<ID=AAC,Number=.,Type=Integer,Description="Counts for all possible alelles">
14 #CHROM  POS ID  REF ALT QUAL    FILTER  INFO    FORMAT  Samplen47_Cleandata_1
15 chrI    114 .   T   C   48  .   CNV=1   GT:GL:GP:GQ:DP:AAC  0/1:-10.43,-2.41,-17.39:0,48,0:48:8:0,3,0,5
16 chrI    115 .   C   A   77  .   CNV=1   GT:GL:GP:GQ:DP:AAC  0/1:-13.91,-3.01,-20.86:0,77,0:77:10:4,6,0,0
17 chrI    136 .   G   A   255 .   CNV=1   GT:GL:GP:GQ:DP:AAC  1:-20.86,-1.81,-0.0:0,0,45:45:6:6,0,0,0
18 chrI    141 .   C   T   166 .   CNV=1   GT:GL:GP:GQ:DP:AAC  0/1:-24.34,-4.52,-27.82:0,255,0:255:15:0,8,0,7
19 chrI    254 .   C   T   48  .   CNV=1   GT:GL:GP:GQ:DP:AAC  0/1:-10.43,-2.41,-17.39:0,48,0:48:8:0,5,0,3
20 chrI    257 .   A   C   109 .   CNV=1   GT:GL:GP:GQ:DP:AAC  0/1:-17.39,-3.31,-20.86:0,109,0:109:11:6,5,0,0
21 chrI    262 .   A   G   255 .   CNV=1   GT:GL:GP:GQ:DP:AAC  1:-66.07,-5.73,-0.01:0,0,84:84:19:0,0,19,0
22 chrI    266 .   T   A   151 .   CNV=1   GT:GL:GP:GQ:DP:AAC  0/1:-24.35,-6.03,-45.21:0,152,0:152:20:7,0,0,13
23 chrI    268 .   A   C   255 .   CNV=1   GT:GL:GP:GQ:DP:AAC  0/1:-34.78,-6.03,-34.78:0,255,0:255:20:10,10,0,0
24 chrI    269 .   C   A   255 .   CNV=1   GT:GL:GP:GQ:DP:AAC  0/1:-38.25,-6.03,-31.3:0,255,0:255:20:11,9,0,0
25 chrI    270 .   C   T   255 .   CNV=1   GT:GL:GP:GQ:DP:AAC  0/1:-34.78,-6.33,-38.25:0,255,0:255:21:0,11,0,10
26 chrI    286 .   A   T   255 .   CNV=1   GT:GL:GP:GQ:DP:AAC  1:-79.97,-6.93,-0.01:0,0,96:96:23:0,0,0,23
27 chrI    291 .   C   T   142 .   CNV=1   GT:GL:GP:GQ:DP:AAC  0/1:-24.35,-6.93,-55.64:0,142,0:142:23:0,16,0,7
28 chrI    303 .   T   C   188 .   CNV=1   GT:GL:GP:GQ:DP:AAC  0/1:-31.3,-9.34,-76.5:0,255,0:255:31:0,9,0,22
29 chrI    305 .   C   G   255 .   CNV=1   GT:GL:GP:GQ:DP:AAC  0/1:-69.55,-12.52,-41.73:0,255,0:255:31:0,11,19,1
30 chrI    308 .   C   T   188 .   CNV=1   GT:GL:GP:GQ:DP:AAC  0/1:-31.3,-9.34,-76.5:0,255,0:255:31:0,22,0,9
31 chrI    313 .   C   T   255 .   CNV=1   GT:GL:GP:GQ:DP:AAC  0/1:-34.78,-10.24,-83.46:0,255,0:255:34:0,24,0,10
32 chrI    349 .   C   T   255 .   CNV=1   GT:GL:GP:GQ:DP:AAC  1:-59.11,-5.12,-0.01:0,0,78:78:17:0,0,0,17
33 chrI    356 .   G   A   97  .   CNV=1   GT:GL:GP:GQ:DP:AAC  0/1:-17.39,-4.52,-34.77:0,97,0:97:15:5,0,10,0
34 chrI    358 .   G   A   255 .   CNV=1   GT:GL:GP:GQ:DP:AAC  1:-52.16,-4.52,-0.01:0,0,72:72:15:15,0,0,0
35 chrI    373 .   T   C   255 .   CNV=1   GT:GL:GP:GQ:DP:AAC  1:-41.73,-3.62,-0.01:0,0,63:63:12:0,12,0,0
36 chrI    476 .   G   C   174 .   CNV=1   GT:GL:GP:GQ:DP:AAC  1:-17.39,-1.51,-0.0:0,0,42:42:5:0,5,0,0
37 chrI    485 .   T   C   174 .   CNV=1   GT:GL:GP:GQ:DP:AAC  1:-17.39,-1.51,-0.0:0,0,42:42:5:0,5,0,0
38 chrI    507 .   C   T   62  .   CNV=1   GT:GL:GP:GQ:DP:AAC  0/1:-13.91,-4.52,-38.25:0,62,0:62:15:0,11,0,4
```

Ilustración 47: archivo VCF generado por ''Find Variants'' con variantes SNPs e Indels.

En esta fila por ejemplo: se puede observar que en la comparación se encontró un variante genómica SNP para el cromosoma número uno del genoma de referencia en la posición 291 donde la muestra tiene un nucleótido Timina en la secuencia de ADN y la referencia tiene un nucleótido Citosina, la siguiente información contenida para ese registro es respecto al genotipo donde se infiere que la variante es un genotipo heterocigoto para la muestra.

Con esta información los biólogos pueden mirar que tan cercanas son las muestras analizadas entre ellas, para mirar por ejemplo relaciones ancestrales y familiares, mirar la diversidad genética de las muestras, la cantidad de heterocigotos, si hay estructuras poblacionales.

Otro archivo generado por el proceso fue el archivo CNV, a diferencia del archivo VCF este archivo contiene solamente el cromosoma donde encontró variantes CNVs, las posiciones de inicio y de fin dentro del genoma de referencia donde está ocurriendo la variante, el número de veces que se repite la variante dentro de diferentes posiciones del genoma e información de calidad, la siguiente imagen muestra una parte del CNV generado por Find Variants para la secuencia de levadura analizada.

Samplen47_Cleandata_1MappingFile.cnv

```
chrI	1	11800	*	*	29	2.53:0:0:0:294
chrI	78	217	*	*	12	0.71:15:0:0:14
chrI	230	1553	*	*	18	1.21:261:4:0:71
chrI	1594	2819	*	*	9	3.61:678:90:0:49
chrI	11801	27100	*	*	255	0.15:0:0:0:51
chrI	12289	12662	*	*	12	0.27:14:0:0:4
chrI	12698	12908	*	*	12	0.26:14:1:0:7
chrI	12957	13256	*	*	14	0.36:23:0:0:1
chrI	24078	25673	*	*	18	0.55:135:2:0:20
chrI	25983	26324	*	*	11	0.18:13:0:0:0
chrI	26609	26920	*	*	12	0.34:16:0:0:6
chrI	27401	27800	*	*	22	0.36:0:0:0:4
chrI	139589	141435	*	*	10	2.04:516:53:0:23
chrI	160201	162300	*	*	226	0.06:0:0:0:0
chrI	163228	163767	*	*	14	0.29:26:0:0:0
chrI	163787	164033	*	*	11	0.24:13:0:0:0
chrI	164263	164427	*	*	8	0.10:5:0:0:0
chrI	165872	165987	*	*	7	0.12:4:0:0:0
chrI	176651	177153	*	*	8	1.82:114:20:0:3
chrI	183901	188700	*	*	85	1.22:0:0:0:63
chrI	187330	189840	*	*	10	2.21:831:104:0:54
chrI	198401	202800	*	*	255	0.10:0:0:0:4
chrI	204001	206300	*	*	110	0.29:0:0:0:50
chrI	204290	204441	*	*	12	0.37:16:0:0:11
chrI	204502	204626	*	*	8	0.20:6:0:0:3
chrI	205674	206104	*	*	16	0.48:43:0:0:25
chrI	206187	208056	*	*	11	2.32:628:60:0:62
chrI	208001	221900	*	*	255	0.19:0:0:0:9
chrI	218596	218836	*	*	11	0.22:13:0:0:0
chrI	219321	220315	*	*	22	0.91:151:0:0:2
chrI	220384	220795	*	*	18	0.83:62:0:0:0
chrI	220814	221421	*	*	20	0.94:92:0:0:3
chrI	221428	223474	*	*	12	3.13:921:56:0:24
chrI	223501	225500	*	*	213	0.03:0:0:0:10
chrI	225413	226896	*	*	28	2.75:624:1:0:8
chrI	226901	230300	*	*	255	0.14:0:0:0:5
chrI	227854	227974	*	*	11	0.42:13:0:0:0
chrI	228074	228377	*	*	14	0.40:24:0:0:3
```

Ilustración 48: archivo CNV generado por ''Find Variants''.

En esta fila por ejemplo: se puede observar que en la comparación se encontró que en el cromosoma uno de la muestra de la posición 204502 a la 204626 hay una secuencia de nucleótidos que se repite ocho veces en diferentes partes del genoma, este tipo de variante conocida como CNV es importante para los biólogos porque pueden.

El siguiente archivo es el GFF, este archivo contiene las variantes CNVs encontradas durante la detección de variantes y los indeles largos.

Samplen47_Cleandata_1MappingFile_SV.gff

```
##gff-version	3
chrI	NGSTools	CNV	1	11800	29	+	.	ID=SV_1;LENGTH=11800;SOURCE=CNVnator;NSF=4797;NC=2.53;HET=294
chrI	NGSTools	REPEAT	78	217	12	+	.	ID=SV_2;LENGTH=140;SOURCE=MultiAlns;NSF=15;NC=0.71;HET=14
chrI	NGSTools	REPEAT	230	1553	18	+	.	ID=SV_3;LENGTH=1324;SOURCE=MultiAlns;NSF=261;NC=1.21;NUF=4;HET=71
chrI	NGSTools	REPEAT	1594	2819	9	+	.	ID=SV_4;LENGTH=1226;SOURCE=MultiAlns;NSF=678;NC=3.61;NUF=90;HET=49
chrI	NGSTools	CNV	11801	27100	255	+	.	ID=SV_5;LENGTH=15300;SOURCE=CNVnator;NSF=366;NC=0.15;HET=51
chrI	NGSTools	REPEAT	12289	12662	12	+	.	ID=SV_6;LENGTH=374;SOURCE=MultiAlns;NSF=14;NC=0.27;HET=4
chrI	NGSTools	REPEAT	12698	12908	12	+	.	ID=SV_7;LENGTH=211;SOURCE=MultiAlns;NSF=14;NC=0.26;NUF=1;HET=7
chrI	NGSTools	REPEAT	12957	13256	14	+	.	ID=SV_8;LENGTH=300;SOURCE=MultiAlns;NSF=23;NC=0.36;HET=1
chrI	NGSTools	REPEAT	24078	25673	18	+	.	ID=SV_9;LENGTH=1596;SOURCE=MultiAlns;NSF=135;NC=0.55;NUF=2;HET=20
chrI	NGSTools	REPEAT	25983	26324	11	+	.	ID=SV_10;LENGTH=342;SOURCE=MultiAlns;NSF=13;NC=0.18;HET=0
chrI	NGSTools	REPEAT	26609	26920	12	+	.	ID=SV_11;LENGTH=312;SOURCE=MultiAlns;NSF=16;NC=0.34;HET=6
chrI	NGSTools	CNV	27401	27800	22	+	.	ID=SV_12;LENGTH=400;SOURCE=CNVnator;NSF=23;NC=0.36;HET=4
chrI	NGSTools	Deletion	51247	51406	21	+	.	ID=SV_13;LENGTH=34;SOURCE=Readpairs;NSF=6
chrI	NGSTools	Deletion	113004	113060	46	+	.	ID=SV_14;LENGTH=36;SOURCE=Readpairs;NSF=10
chrI	NGSTools	REPEAT	139589	141435	10	+	.	ID=SV_15;LENGTH=1847;SOURCE=MultiAlns;NSF=516;NC=2.04;NUF=53;HET=23
chrI	NGSTools	CNV	160201	162300	255	+	.	ID=SV_16;LENGTH=2100;SOURCE=CNVnator;NSF=20;NC=0.06;HET=0
chrI	NGSTools	Deletion	160203	166205	160	+	.	ID=SV_17;LENGTH=5935;SOURCE=Readpairs;NSF=53
chrI	NGSTools	REPEAT	163228	163767	14	+	.	ID=SV_18;LENGTH=540;SOURCE=MultiAlns;NSF=26;NC=0.29;HET=0
chrI	NGSTools	REPEAT	163787	164033	11	+	.	ID=SV_19;LENGTH=247;SOURCE=MultiAlns;NSF=13;NC=0.24;HET=0
chrI	NGSTools	REPEAT	164263	164427	8	+	.	ID=SV_20;LENGTH=165;SOURCE=MultiAlns;NSF=5;NC=0.1;HET=0
chrI	NGSTools	REPEAT	165872	165987	7	+	.	ID=SV_21;LENGTH=116;SOURCE=MultiAlns;NSF=4;NC=0.12;HET=0
chrI	NGSTools	REPEAT	176651	177153	8	+	.	ID=SV_22;LENGTH=503;SOURCE=MultiAlns;NSF=114;NC=1.82;NUF=20;HET=3
chrI	NGSTools	Deletion	182512	182997	95	+	.	ID=SV_23;LENGTH=350;SOURCE=Readpairs;NSF=16
chrI	NGSTools	CNV	183901	188700	85	+	.	ID=SV_24;LENGTH=4800;SOURCE=CNVnator;NSF=939;NC=1.22;HET=63
chrI	NGSTools	Insertion	184620	187104	28	+	.	ID=SV_25;LENGTH=472;SOURCE=Readpairs;NSF=3
chrI	NGSTools	REPEAT	187330	189840	10	+	.	ID=SV_26;LENGTH=2511;SOURCE=MultiAlns;NSF=831;NC=2.21;NUF=104;HET=54
chrI	NGSTools	CNV	198401	202800	255	+	.	ID=SV_27;LENGTH=4400;SOURCE=CNVnator;NSF=74;NC=0.1;HET=4
chrI	NGSTools	CNV	204001	206300	110	+	.	ID=SV_28;LENGTH=2300;SOURCE=CNVnator;NSF=107;NC=0.29;HET=50
chrI	NGSTools	REPEAT	204290	204441	12	+	.	ID=SV_29;LENGTH=152;SOURCE=MultiAlns;NSF=16;NC=0.37;HET=11
chrI	NGSTools	REPEAT	204502	204626	8	+	.	ID=SV_30;LENGTH=125;SOURCE=MultiAlns;NSF=6;NC=0.2;HET=3
chrI	NGSTools	REPEAT	205674	206104	16	+	.	ID=SV_31;LENGTH=431;SOURCE=MultiAlns;NSF=43;NC=0.48;HET=25
chrI	NGSTools	REPEAT	206187	208056	11	+	.	ID=SV_32;LENGTH=1870;SOURCE=MultiAlns;NSF=628;NC=2.32;NUF=60;HET=62
chrI	NGSTools	CNV	208001	221900	255	+	.	ID=SV_33;LENGTH=13900;SOURCE=CNVnator;NSF=426;NC=0.19;HET=9
chrI	NGSTools	REPEAT	218596	218836	11	+	.	ID=SV_34;LENGTH=241;SOURCE=MultiAlns;NSF=13;NC=0.22;HET=0
chrI	NGSTools	REPEAT	219321	220315	22	+	.	ID=SV_35;LENGTH=995;SOURCE=MultiAlns;NSF=151;NC=0.91;HET=2
chrI	NGSTools	REPEAT	220384	220795	18	+	.	ID=SV_36;LENGTH=412;SOURCE=MultiAlns;NSF=62;NC=0.83;HET=0
chrI	NGSTools	REPEAT	220814	221421	20	+	.	ID=SV_37;LENGTH=608;SOURCE=MultiAlns;NSF=92;NC=0.94;HET=3
```

Ilustración 49: archivo GFF generado por ''Find Variants''.

En esta fila por ejemplo: se encontró un indel largo que para este caso es una deleción o eliminación de nucleótidos en la secuencia de ADN de la posición 182512 a la 182997 de 485 nucleótidos.

Por último, se encuentra el archivo de historial de Find Variants Ilustración 50, que almacena la muestra, el genoma de referencia y el resultado de la detección de variantes.



Ilustración 50: archivo de historial con la última muestra, genoma de referencia y archivo vcf de salida generado por ''Find Variants''.

4.3.5 ANOTACIÓN DE GENES

Después de finalizar el proceso de detección de variantes en el cuarto proceso, se procede a verificar si las variantes encontradas tienen algún tipo de influencia en la función de los genes, esta verificación para esta prueba es el resultado de comparar un catálogo de genes del genoma de la levadura, el genoma de levadura, y el archivo VCF con variantes genómicas detectadas por Find Variants para la muestra Sample47 de levadura.

Para acceder a esta función se repiten los pasos anteriores, se ubica el archivo VCF "Samplen47_Cleandata_1MappingFile.vcf" luego se da clic derecho en este, y se busca la opción Variants Functional Annotation dentro de NGSEP Menu.

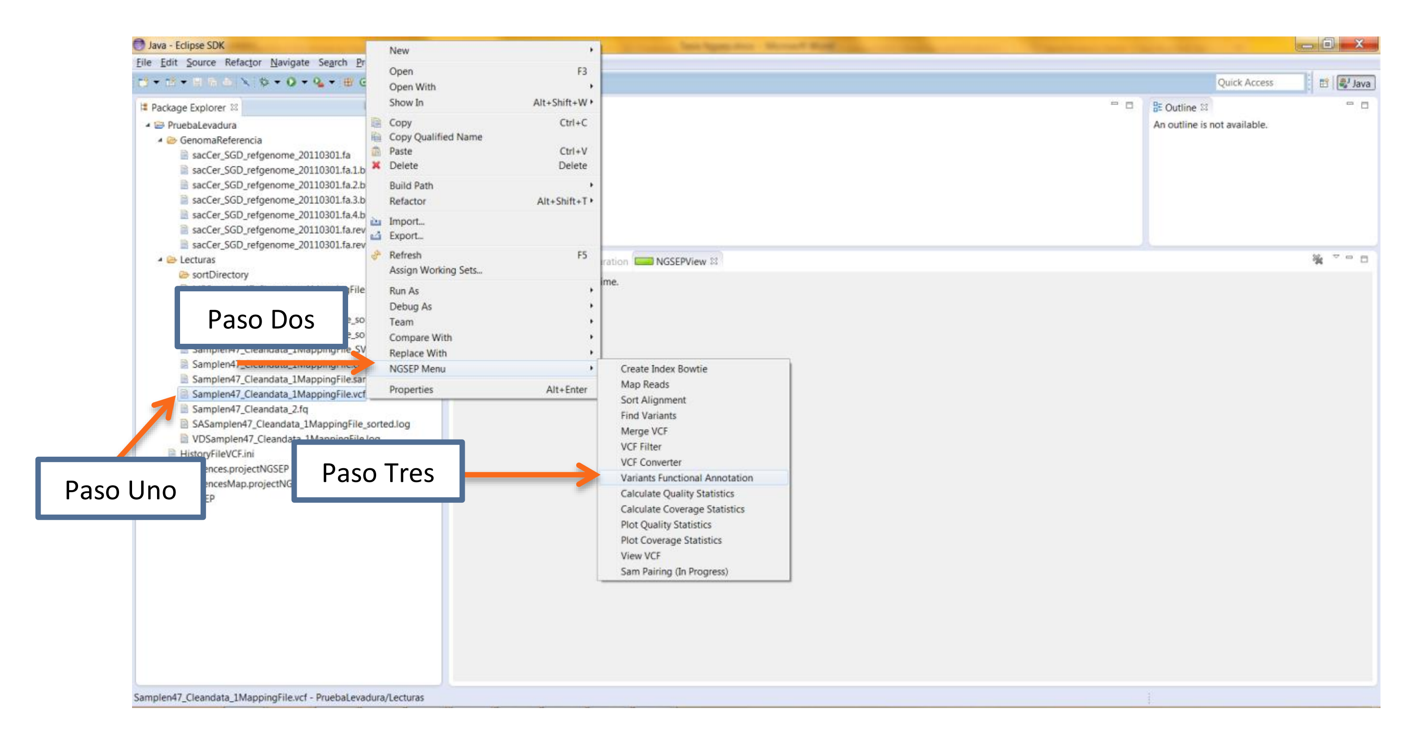


Ilustración 51: accediendo a ''Variants Functional Annotation''.

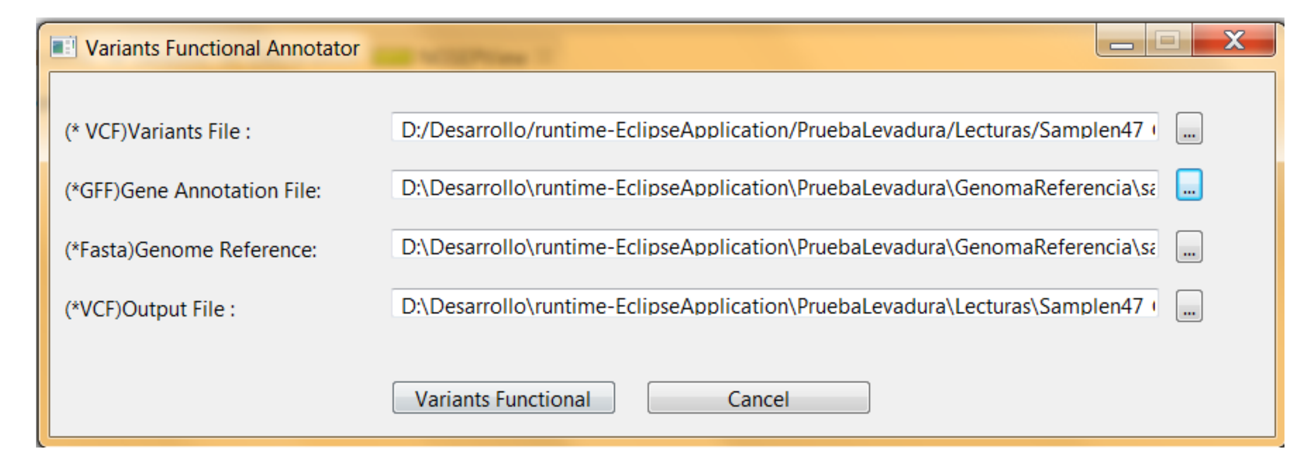


Ilustración 52: pantalla de ''Variants Functional Annotation''.

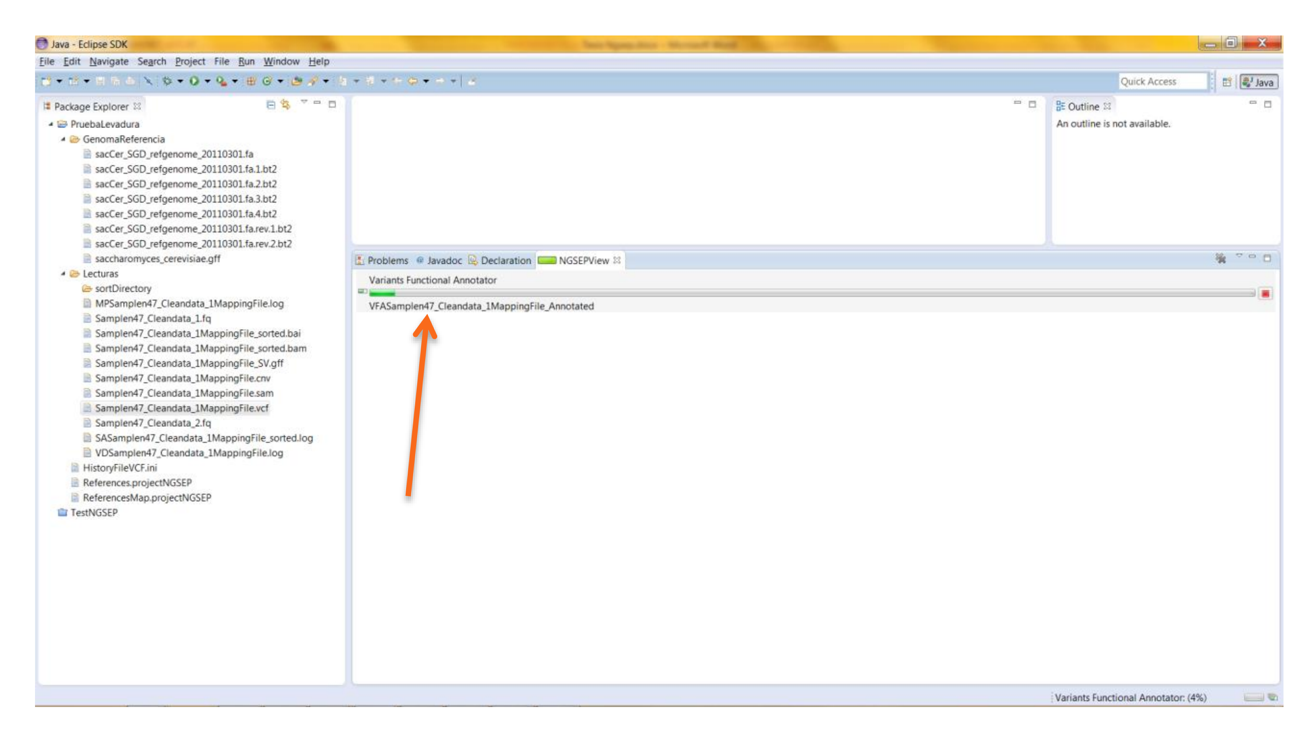


Ilustración 53: barra de progreso generada por ''Variants Functional Annotator''.

El proceso de ejecución lleva a cabo la comparación entre el genoma de levadura y la muestra o lectura Sample47 que es una extracción de la secuenciada de una planta de levadura, y un catálogo de genes del genoma de levadura, esta comparación va arrojar un archivo VCF con todas las variantes genómicas encontradas durante la comparación posición a posición del genoma de referencia y su efecto en los genes de la levadura.

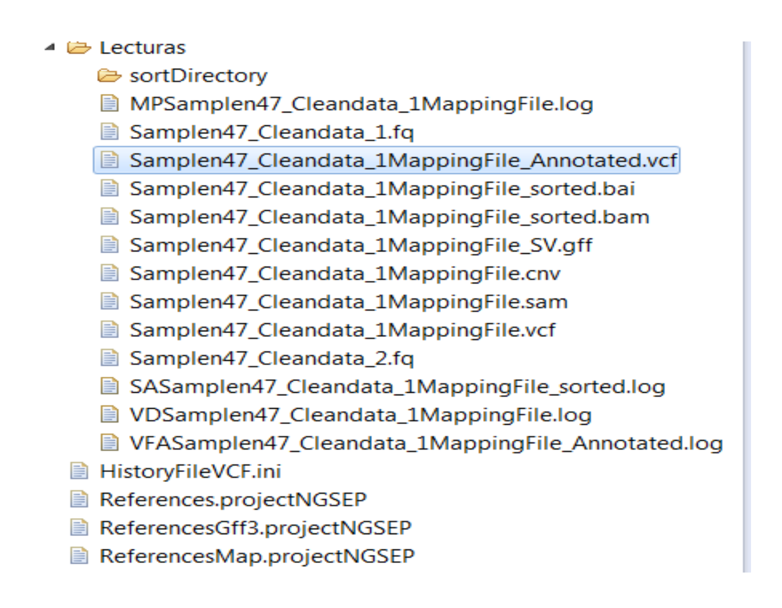


Ilustración 54: Archivo generado por 'Variants Functional Annotator'.

El resultado de la ejecución de Functional Variants Annotation arroja el archivo “Samplen47_Cleandata_1MappingFile_Annotated.vcf”.

```
Samplen47_Cleandata_1MappingFile_Annotated.vcf
##fileformat=VCFv4.1
##INFO=<ID=CNV,Number=1,Type=Integer,Description="Number of samples with CNVs around this variant">
##INFO=<ID=TA,Number=1,Type=String,Description="Variant annotation based on a gene model">
##INFO=<ID=TID,Number=1,Type=String,Description="Id of the transcript related to the variant annotation">
##INFO=<ID=TGN,Number=1,Type=String,Description="Name of the gene related to the variant annotation">
##INFO=<ID=TCO,Number=1,Type=Float,Description="One based codon position of the start of the variant. The decimal is the codon position">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GL,Number=G,Type=Float,Description="Genotype likelihoods">
##FORMAT=<ID=GP,Number=G,Type=Integer,Description="Genotype posterior probabilities">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype quality">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read depth">
##FORMAT=<ID=AC,Number=A,Type=Integer,Description="Counts for observed alleles">
##FORMAT=<ID=AAC,Number=.,Type=Integer,Description="Counts for all possible alelles">
#CHROM  POS ID  REF ALT QUAL   FILTER  INFO    FORMAT  Samplen47_Cleandata_1
chrI    114 .   T   C   48   .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  0/1:-10.43,-2.41,-17.39:0,48,0:48:8:0,3,0,5
chrI    115 .   C   A   77   .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  0/1:-13.91,-3.01,-20.86:0,77,0:77:10:4,6,0,0
chrI    136 .   G   A   255  .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  1:-20.86,-1.81,-0.0:0,0,45:45:6:6,0,0,0
chrI    141 .   C   T   166  .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  0/1:-24.34,-4.52,-27.82:0,255,0:255:15:0,8,0,7
chrI    254 .   C   T   48   .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  0/1:-10.43,-2.41,-17.39:0,48,0:48:8:0,5,0,3
chrI    257 .   A   C   109  .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  0/1:-17.39,-3.31,-20.86:0,109,0:109:11:6,5,0,0
chrI    262 .   A   G   255  .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  1:-66.07,-5.73,-0.01:0,0,84:84:19:0,0,19,0
chrI    266 .   T   A   151  .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  0/1:-24.35,-6.03,-45.21:0,152,0:152:20:7,0,0,13
chrI    268 .   A   C   255  .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  0/1:-34.78,-6.03,-34.78:0,255,0:255:20:10,10,0,0
chrI    269 .   C   A   255  .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  0/1:-38.25,-6.03,-31.3:0,255,0:255:20:11,9,0,0
chrI    270 .   C   T   255  .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  0/1:-34.78,-6.33,-38.25:0,255,0:255:21:0,11,0,10
chrI    286 .   A   T   255  .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  1:-79.97,-6.93,-0.01:0,0,96:96:23:0,0,0,23
chrI    291 .   C   T   142  .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  0/1:-24.35,-6.93,-55.64:0,142,0:142:23:0,16,0,7
chrI    303 .   T   C   188  .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  0/1:-31.3,-9.34,-76.5:0,255,0:255:31:0,9,0,22
chrI    305 .   C   G   255  .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  0/1:-69.55,-12.52,-41.73:0,255,0:255:31:0,11,19,1
chrI    308 .   C   T   188  .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  0/1:-31.3,-9.34,-76.5:0,255,0:255:31:0,22,0,9
chrI    313 .   C   T   255  .   CNV=1;TA=Upstream;TGN=YAL068W-A;TID=YAL068W-A_mRNA  GT:GL:GP:GQ:DP:AAC  0/1:-34.78,-10.24,-83.46:0,255,0:255:34:0,24,0,10
chrI    349 .   C   T   255  .   CNV=1;TA=Synonymous;TCO=5.3;TGN=YAL069W;TID=YAL069W_mRNA    GT:GL:GP:GQ:DP:AAC  1:-59.11,-5.12,-0.01:0,0,78:78:17:0,0,0,17
chrI    356 .   G   A   97   .   CNV=1;TA=Missense;TCO=8.1;TGN=YAL069W;TID=YAL069W_mRNA  GT:GL:GP:GQ:DP:AAC  0/1:-17.39,-4.52,-34.77:0,97,0:97:15:5,0,10,0
chrI    358 .   G   A   255  .   CNV=1;TA=Synonymous;TCO=8.3;TGN=YAL069W;TID=YAL069W_mRNA    GT:GL:GP:GQ:DP:AAC  1:-52.16,-4.52,-0.01:0,0,72:72:15:15,0,0,0
chrI    373 .   T   C   255  .   CNV=1;TA=Synonymous;TCO=13.3;TGN=YAL069W;TID=YAL069W_mRNA   GT:GL:GP:GQ:DP:AAC  1:-41.73,-3.62,-0.01:0,0,63:63:12:0,12,0,0
chrI    476 .   G   C   174  .   CNV=1;TA=Missense;TCO=48.1;TGN=YAL069W;TID=YAL069W_mRNA GT:GL:GP:GQ:DP:AAC  1:-17.39,-1.51,-0.0:0,0,42:42:5:0,5,0,0
chrI    485 .   T   C   174  .   CNV=1;TA=Missense;TCO=51.1;TGN=YAL069W;TID=YAL069W_mRNA GT:GL:GP:GQ:DP:AAC  1:-17.39,-1.51,-0.0:0,0,42:42:5:0,5,0,0
chrI    507 .   C   T   62   .   CNV=1;TA=Missense;TCO=58.2;TGN=YAL069W;TID=YAL069W_mRNA GT:GL:GP:GQ:DP:AAC  0/1:-13.91,-4.52,-38.25:0,62,0:62:15:0,11,0,4
```

Ilustración 55: archivo vcf con variantes y la región donde fue encontrada la variante.

En esta fila del archivo de anotación de genes: se encuentra una variante genómica de tipo SNP en el cromosoma uno en la posición 141 del genoma de levadura, esta variante se detectó en una región codificante de gen, se denominan Upstream a las regiones codificantes de genes dentro del genoma de referencia.

En este sentido, es importante para los biólogos conocer que tipo influencia tienen las variantes genómicas dentro de una secuencia de ADN porque esto les permite diagnosticar e identificar genes para determinada especie.

4.3.6 MEZCLAR VCFS

Este proceso es el encargado de mezclar la información con respecto a variantes genómicas de varias muestras que compartan relación genética, con el fin de generar un solo archivo VCF con dicha información.

Para acceder a este proceso se debe seleccionar el historial de variants detector, luego dar clic derecho encima de la selección y buscar dentro de “NSGEP Menu” la opción “Merge VCF”.

Antes de comenzar este proceso, se tienen que tener por lo menos tres muestras con información genética relacionada, por ejemplo el caso de dos papás y un hijo de levadura. Con estas muestras se debe proceder a realizar el cuarto proceso de detección de variantes en caso de que se encuentren ya secuenciadas y alineadas con

respecto al genoma de referencia de levadura en un archivo BAM, sino se debe comenzar desde el primer proceso con las lecturas crudas.

En este ejemplo se van utilizar las siguientes muestras de levadura:

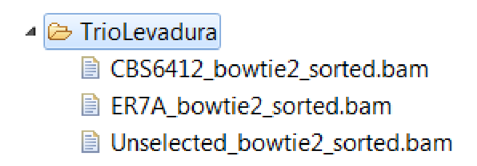


Ilustración 56: archivos usados para ejecutar ''Merge VCF''.

Siendo "CBS6412_bowtie2_sorted.bam" y "ER7A_bowtie2_sorted.bam" los papas de "Unselected_bowtie2_sorted.bam".

Primer paso:

- ✓ Detección de variantes genómicas para la muestra papa "CBS6412".
- ✓ Detección de variantes genómicas para la muestra papa "ER7A".
- ✓ Detección de variantes genómicas para la muestra hijo "Unselected".

4.3.6.1 DETECCIÓN DE VARIANTES PARA LA MUESTRA DE LEVADURA PAPÁ "CBS6412_bowtie2_sorted.bam".

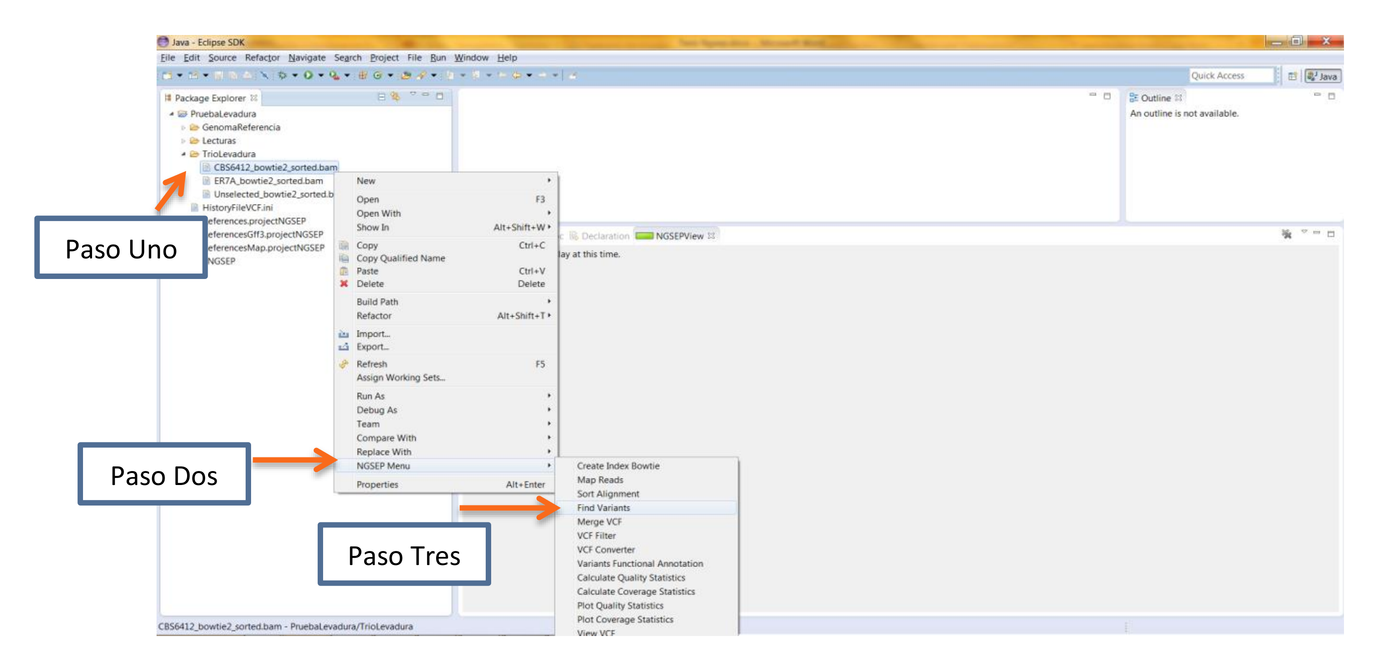


Ilustración 57: accediendo a ''Find Variants'' con la muestra "CBS6412_bowtie2_sorted.bam".

4.3.6.2 DETECCIÓN DE VARIANTES PARA LA MUESTRA DE LEVADURA PAPÁ “ER7A _bowtie2_sorted.bam”.

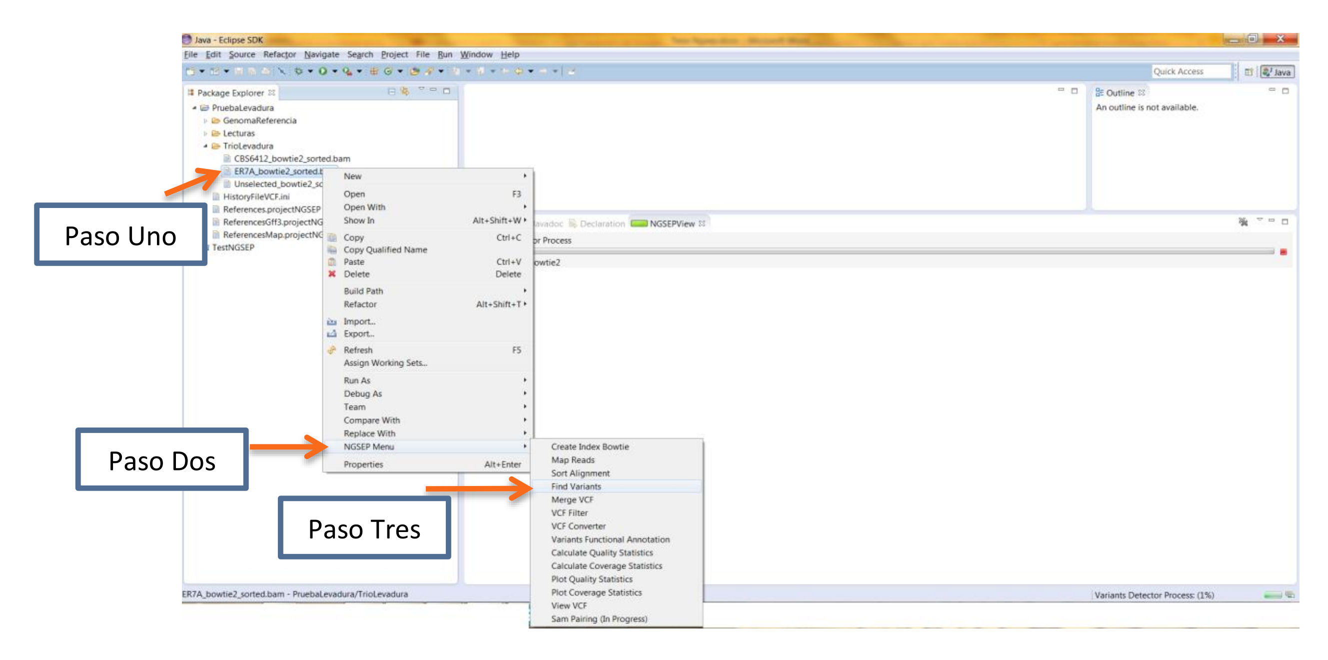


Ilustración 58: accediendo a ‘’Find Variants’’ con la muestra “ER7A _bowtie2_sorted.bam”.

4.3.6.3 DETECCIÓN DE VARIANTES PARA LA MUESTRA DE LEVADURA HIJO “Unselected _bowtie2_sorted.bam”.

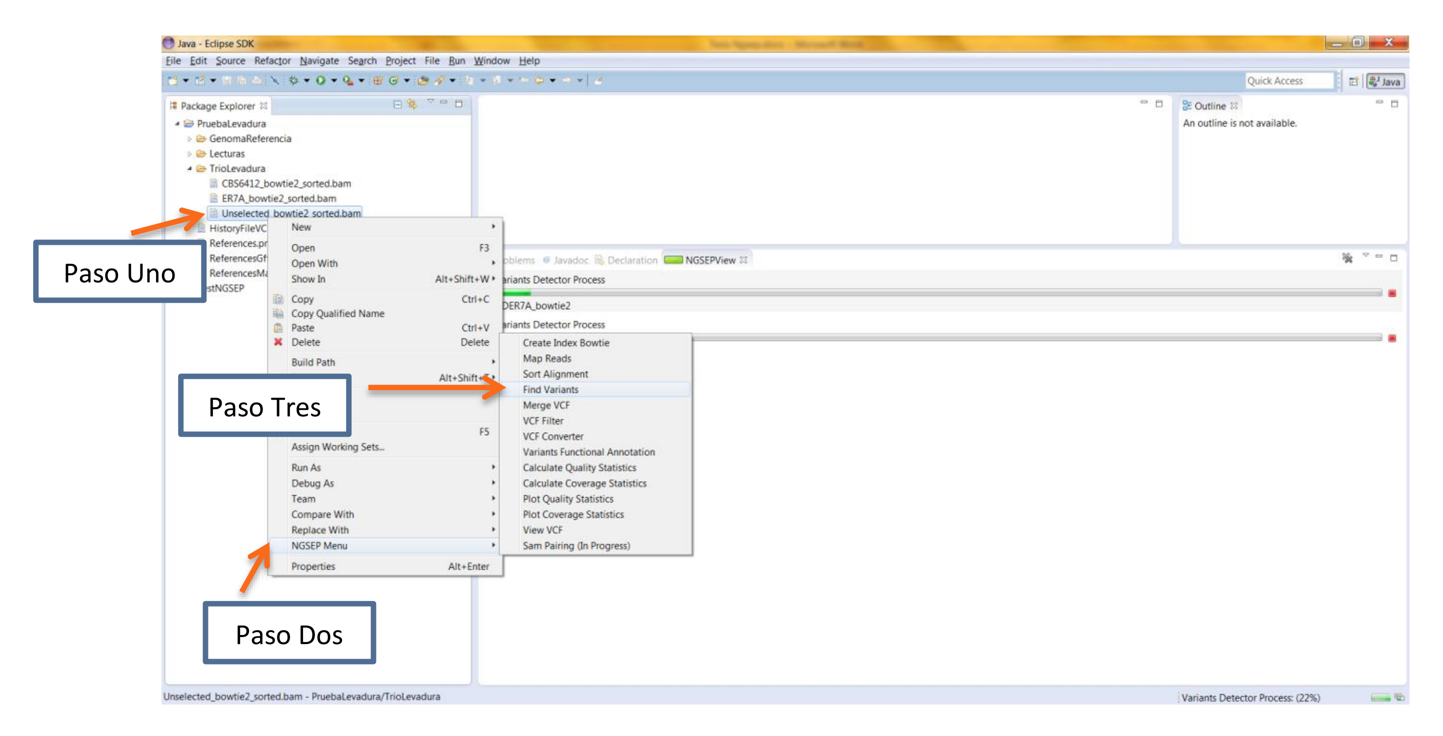


Ilustración 59: accediendo a ‘’Find Variants’’ con la muestra “Unselected _bowtie2_sorted.bam”.

Ejecución de Find Variants para las tres muestras.

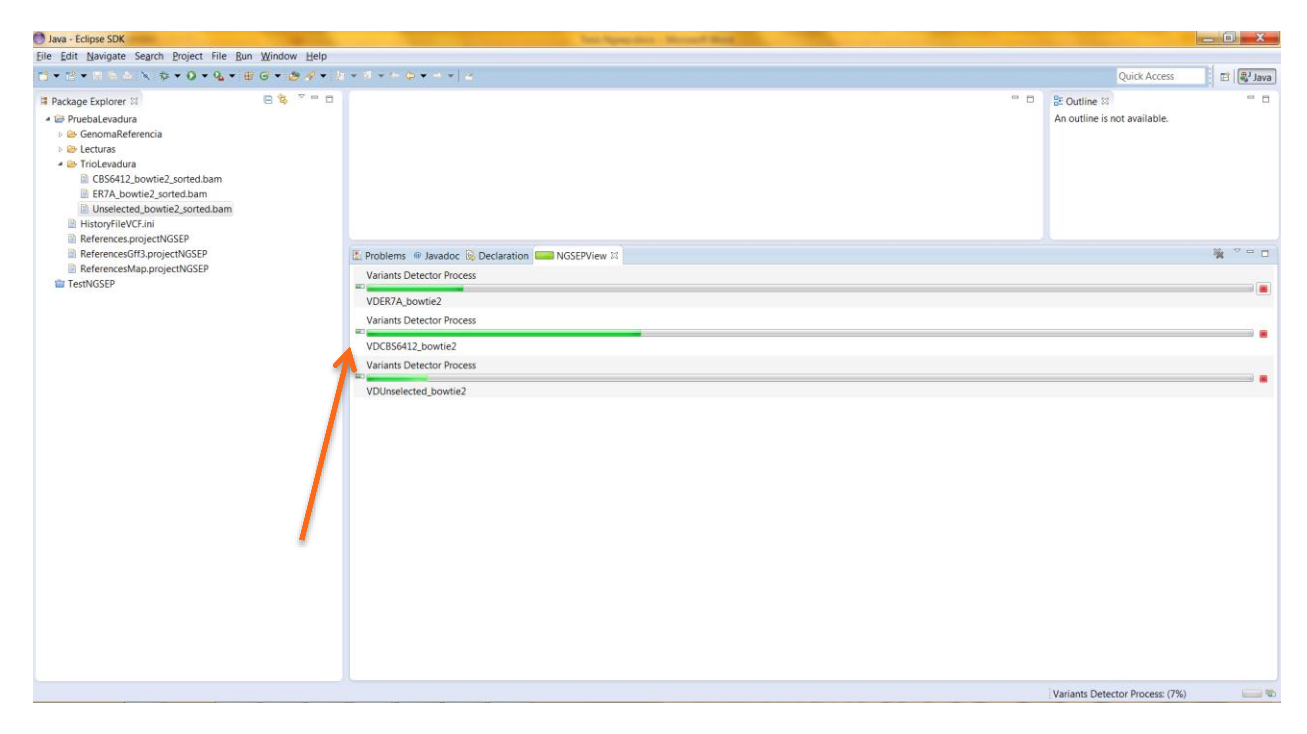


Ilustración 60: ejecución de ''Find Variants'' con las tres muestras.

A continuación, se accede por primera vez a la pantalla de "Merge VCF", para acceder a este proceso se debe seleccionar el historial de Find Variants con las tres muestras ya ejecutadas.

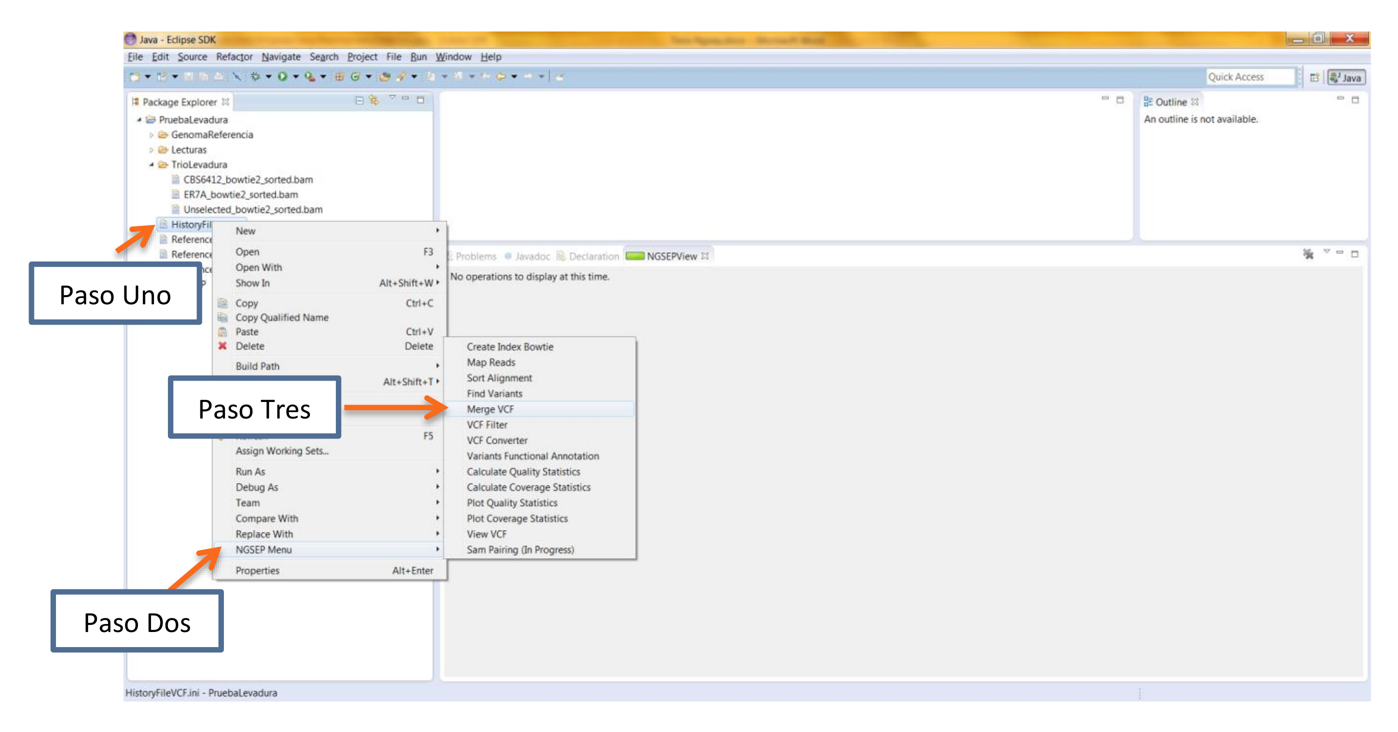


Ilustración 61: accediendo a ''Merge VCF''.

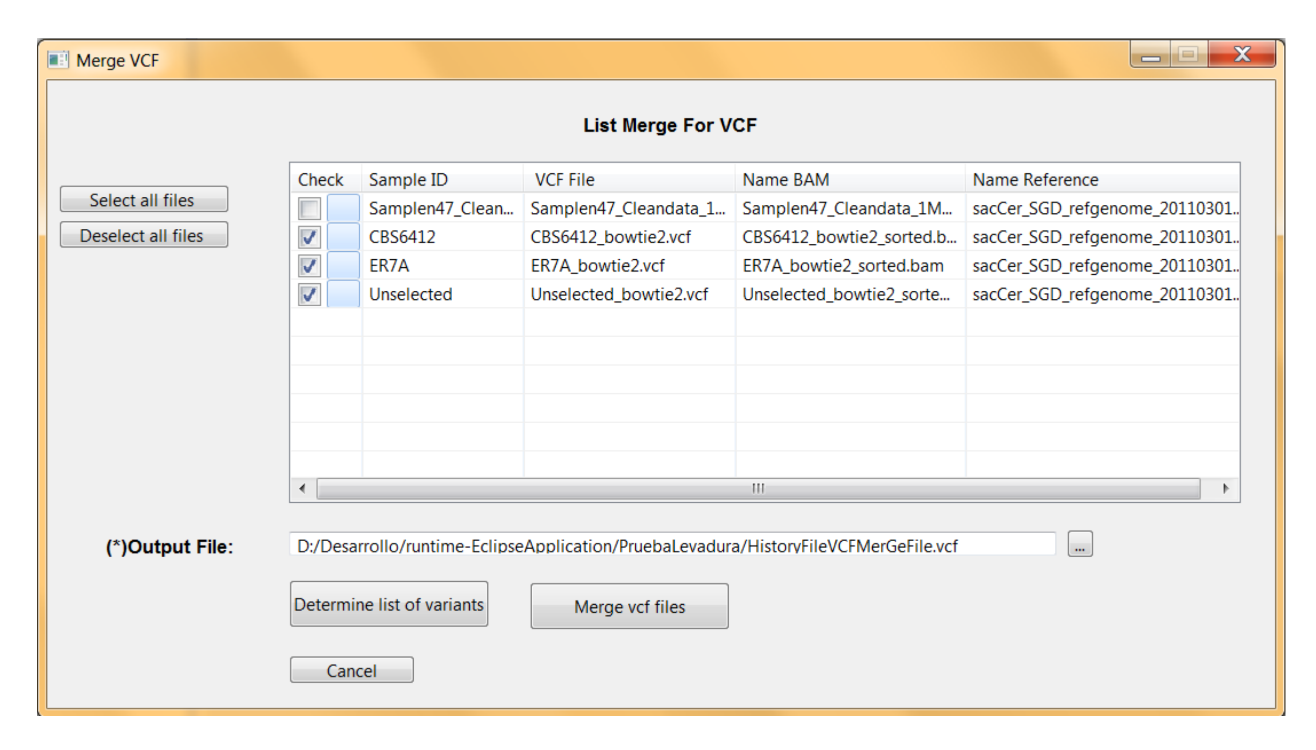


Ilustración 62: pantalla de ''Merge VCF''.

En la imagen anterior ya se encuentran seleccionados las tres muestras que estamos utilizando para este ejemplo, una vez seleccionados los archivos a los que se quiere hacer Merge, se procede a determinar un listado de variantes comunes entre las tres muestras y registrarlas en un solo archivo VCF, esto con el fin de facilitar su análisis desde ancestros hasta los descendientes, para realizar este proceso se debe hacer clic en el botón Determine list of variants.

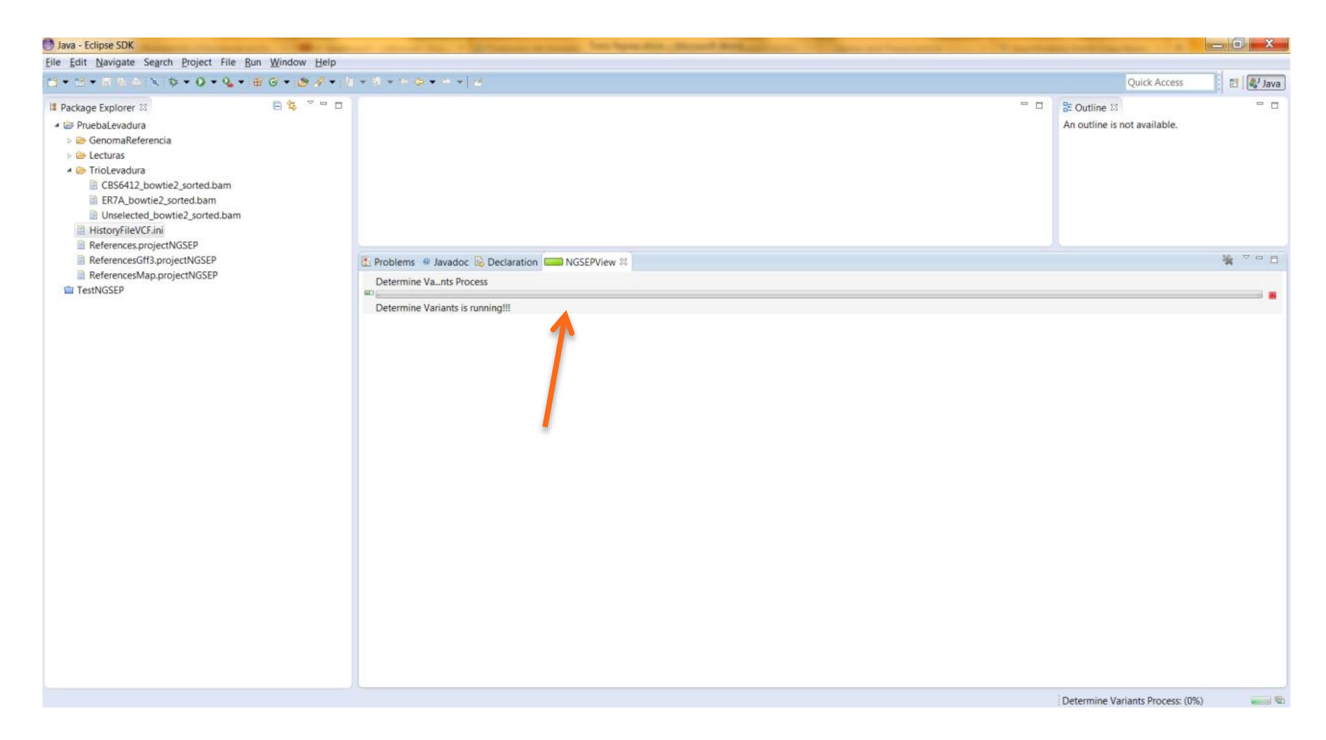
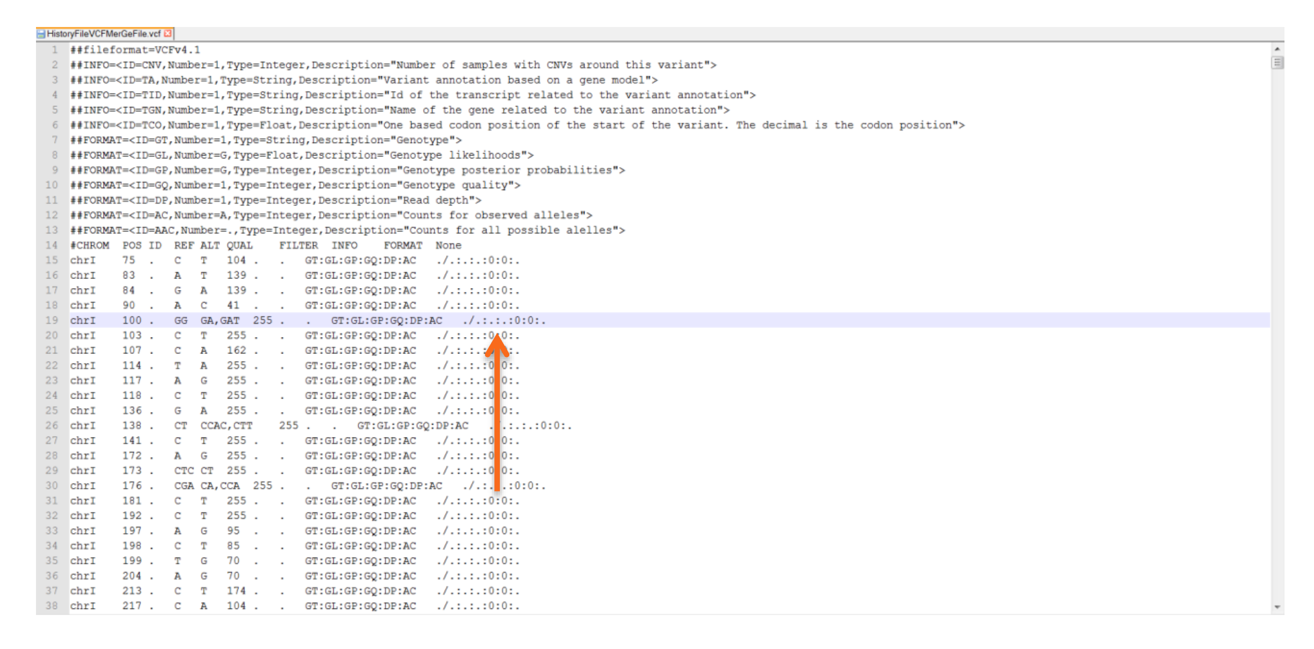


Ilustración 63: ejecutando la opción ''determine list of variants'' dentro del proceso ''Merge VCF''.

Luego de finalizar el proceso para determinar variantes, se genera un archivo VCF de nombre ''HistoryFileVCFMerGeFile.vcf'', después se debe volver a correr el proceso de Find Variants para las tres muestras que se seleccionaron en el proceso para determinar variantes, estas tres muestras son los dos papás y el hijo, pero esta vez la ejecución de Find Variants tiene un parámetro de entrada nuevo que es el archivo de variantes comunes (''HistoryFileVCFMerGeFile.vcf''), este archivo se debe cargar en la opción "Known Variants File:".



```
HistoryFileVCFMerGeFile.vcf
1  ##fileformat=VCFv4.1
2  ##INFO=<ID=CNV,Number=1,Type=Integer,Description="Number of samples with CNVs around this variant">
3  ##INFO=<ID=TA,Number=1,Type=String,Description="Variant annotation based on a gene model">
4  ##INFO=<ID=TID,Number=1,Type=String,Description="Id of the transcript related to the variant annotation">
5  ##INFO=<ID=TGN,Number=1,Type=String,Description="Name of the gene related to the variant annotation">
6  ##INFO=<ID=TCO,Number=1,Type=Float,Description="One based codon position of the start of the variant. The decimal is the codon position">
7  ##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
8  ##FORMAT=<ID=GL,Number=G,Type=Float,Description="Genotype likelihoods">
9  ##FORMAT=<ID=GP,Number=G,Type=Integer,Description="Genotype posterior probabilities">
10 ##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype quality">
11 ##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read depth">
12 ##FORMAT=<ID=AC,Number=A,Type=Integer,Description="Counts for observed alleles">
13 ##FORMAT=<ID=AAC,Number=.,Type=Integer,Description="Counts for all possible alelles">
14 #CHROM  POS ID  REF ALT QUAL    FILTER  INFO    FORMAT  None
15 chrI    75  .   C   T   104 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
16 chrI    83  .   A   T   139 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
17 chrI    84  .   G   A   139 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
18 chrI    90  .   A   C   41  .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
19 chrI    100 .   GG  GA,GAT  255 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
20 chrI    103 .   C   T   255 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
21 chrI    107 .   C   A   162 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
22 chrI    114 .   T   A   255 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
23 chrI    117 .   A   G   255 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
24 chrI    118 .   C   T   255 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
25 chrI    136 .   G   A   255 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
26 chrI    138 .   CT  CCAC,CTT    255 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
27 chrI    141 .   C   T   255 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
28 chrI    172 .   A   G   255 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
29 chrI    173 .   CTC CT  255 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
30 chrI    176 .   CGA CA,CCA  255 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
31 chrI    181 .   C   T   255 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
32 chrI    192 .   C   T   255 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
33 chrI    197 .   A   G   95  .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
34 chrI    198 .   C   T   85  .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
35 chrI    199 .   T   G   70  .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
36 chrI    204 .   A   G   70  .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
37 chrI    213 .   C   T   174 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
38 chrI    217 .   C   A   104 .   .   GT:GL:GP:GQ:DP:AC   ./.:.:.:0:0:.
```

Ilustración 64: archivo VCF con las variantes comunes de las tres muestras.

Ilustración 64: archivo de variantes comunes entre las muestras "CBS6412….", "ER7A….." y "Unselected…." Sin información de genotipos por variante.

4.3.6.4 DETECCIÓN DE VARIANTES PARA LAS MUESTRAS SELECCIONADAS EN MERGE.

Esta nueva ejecución de Find Variants Ilustración 65, para cada muestra con la adicción del archivo de variantes comunes tiene como fin generar un nuevo archivo VCF con las variantes comunes para la muestra que se esté ejecutando con información del genotipo de dicha variante.

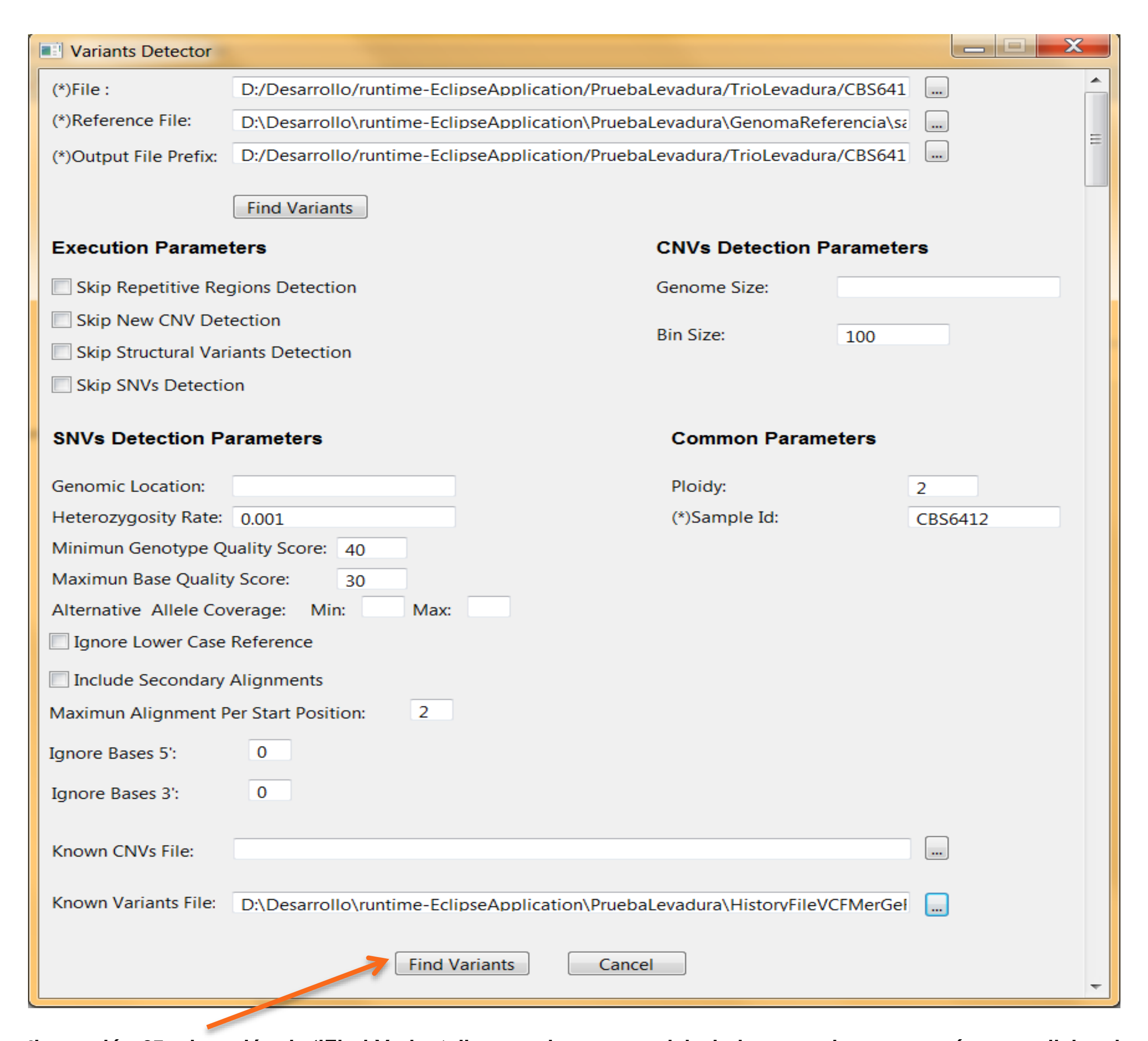


Ilustración 65: ejecución de ''Find Variants'' por cada muestra del trio ingresando como parámetro adicional el archivo con la lista de variantes comunes entre las tres muestras.

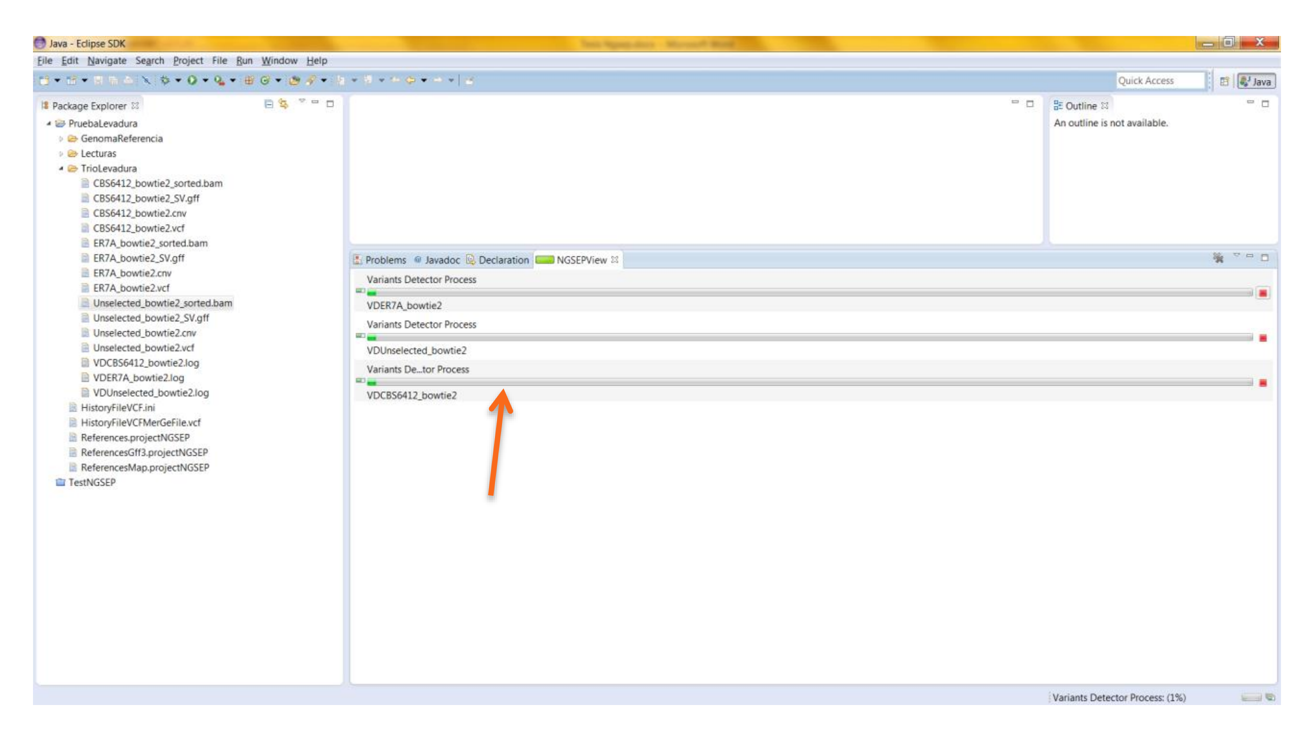


Ilustración 66: ejecución de ''Find Variants'' por cada muestra con el archivo VCF de variantes comunes.

Ejecución de Find Variants Ilustración 66, para las tres muestras con el archivo común de variantes, cuando finaliza los procesos de Find Variants para las tres muestras, se puede proseguir con el proceso final de Merge VCF.

Para iniciar el proceso final de Merge VCF Ilustración 67, se debe seleccionar de nuevo el historial de Find Variants “”HistoryFileVCF.Ini” e ingresar a Merge VCF en las opciones de NGSEP Menu.

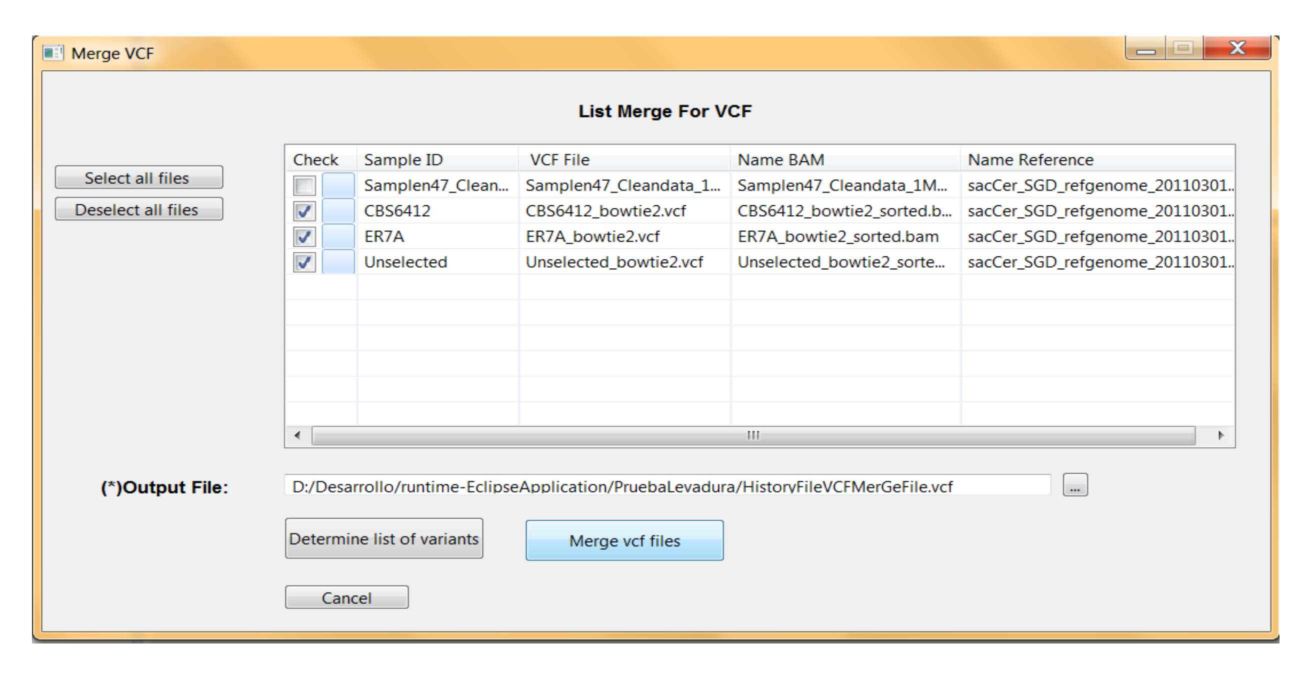


Ilustración 67: Pantalla de ''Merge VCF'' con las nuevos VCFs.

Se seleccionan las mismas muestras que fueron seleccionadas para determinar el archivo de variantes comunes y se da clic en el botón Merge VCF files.

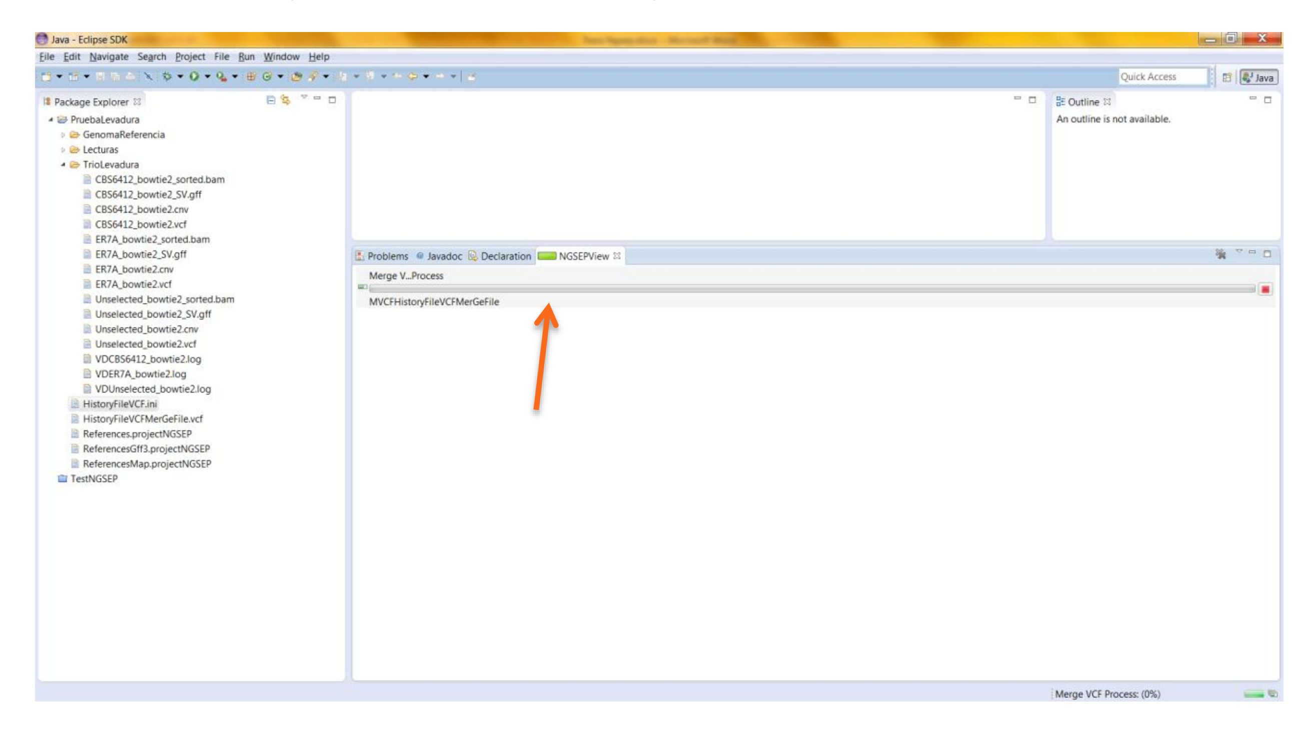


Ilustración 68: ejecución de la opción ‘’’Merge VCF Files’’ del proceso ‘’Merge VCF’’.

Este proceso va generar un archivo VCF con todas las variantes genómicas entre las muestras seleccionadas con su respectivo genotipo por muestra.

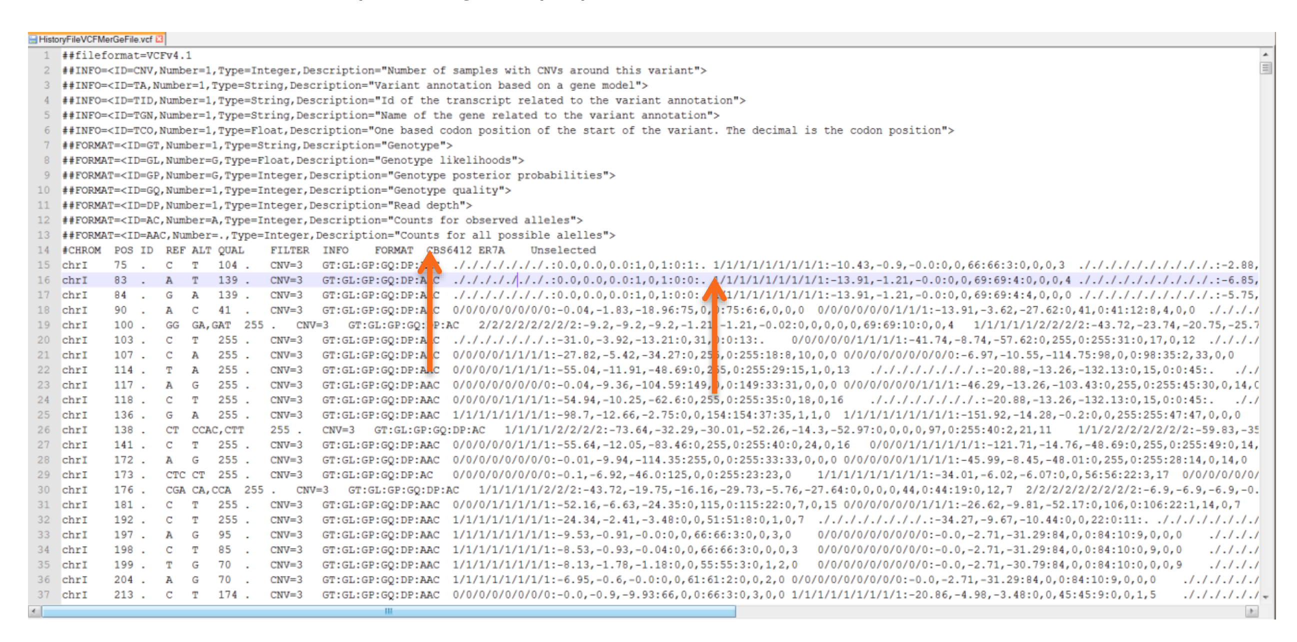


Ilustración 69: archivo VCF con cada una de las muestras y sus variantes comunes con el respectivo genotipo.

Este proceso tiene como fin conocer la herencia de variantes genómicas entre descendientes y ancestros.

4.3.7 CALCULAR ESTADISTICAS DE CALIDAD

Este proceso es el encargado de comparar el archivo BAM, con el genoma de referencia de yuca para esta prueba se procede a indicar el número de errores de secuenciación si lo hay para cada posición de las lecturas alineadas. Se debe tener una distribución homogénea alrededor de cada lectura, para esta prueba se continua utilizando la muestra Sample47 que es con la que se realizó la mayoría de los procesos anteriores.

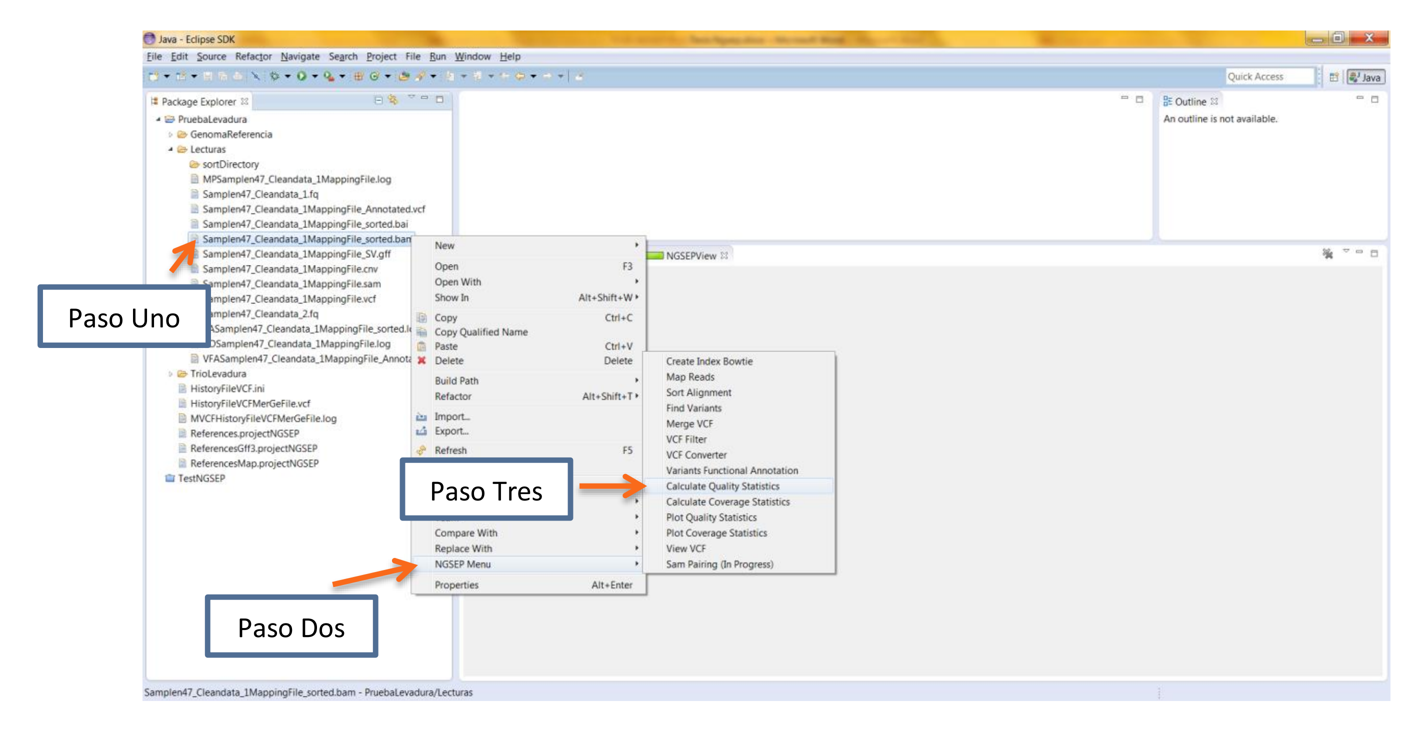


Ilustración 70: Accediendo a ''Calculate Quality Statistics''.

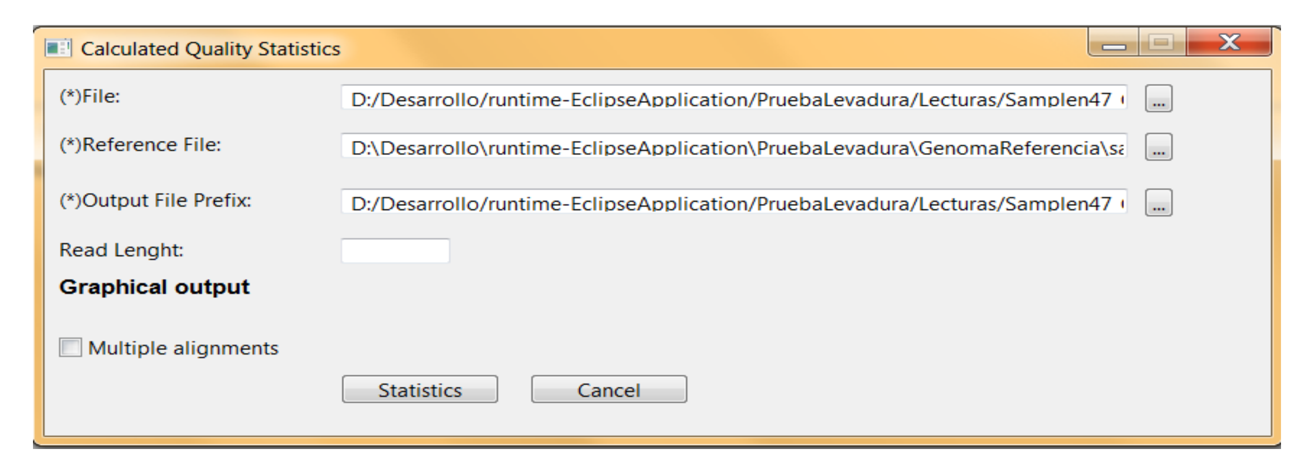


Ilustración 71: pantalla de ''calculate Quality Statistics''.

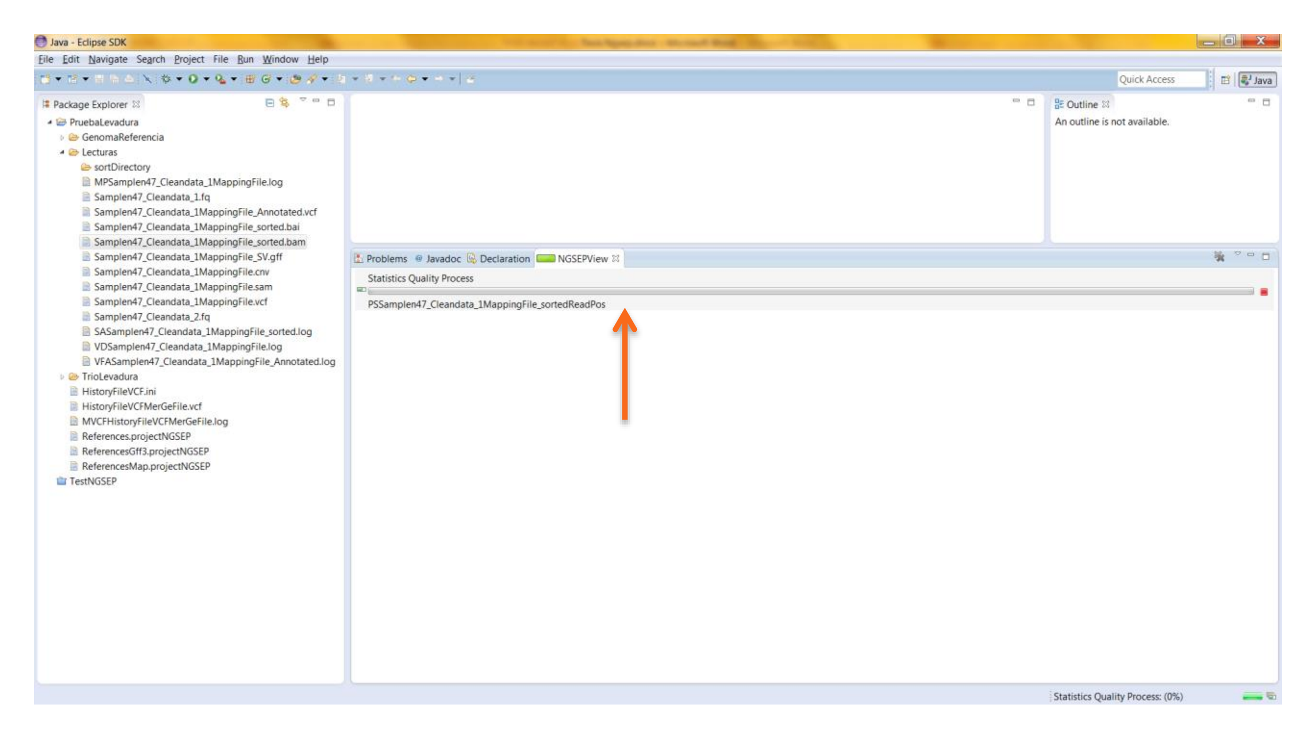


Ilustración 72: ejecución de ''calculate Quality Statistics''.

Al finalizar la ejecución de Quality Statistics, se generan un archivo con las estadísticas y otro con la gráfica con la calidad de la secuencia presente en el BAM.

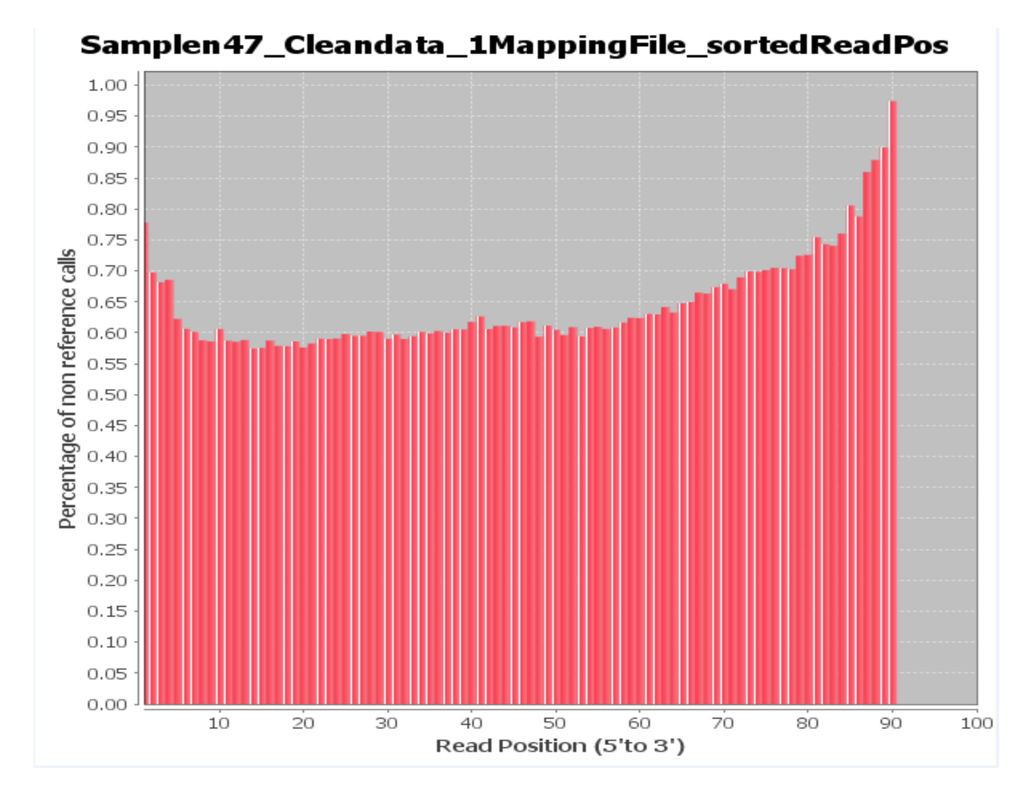


Ilustración 73: Grafica de ''calculate Quality Statistics''.

Samplen47_Cleandata_1MappingFile_sortedReadPos.stats

1	1	35974	29622
2	2	32245	26542
3	3	31556	25949
4	4	31392	26101
5	5	28183	23699
6	6	27190	23078
7	7	26855	22896
8	8	26215	22371
9	9	26131	22306
10	10	26990	23079
11	11	25987	22349
12	12	25988	22284
13	13	25924	22390
14	14	25445	21873
15	15	25306	21907
16	16	25759	22359
17	17	25437	22030
18	18	25502	22016
19	19	25750	22309
20	20	25364	21934
21	21	25615	22188
22	22	25911	22477
23	23	25983	22461
24	24	26067	22484
25	25	26273	22758
26	26	26083	22645
27	27	26049	22654
28	28	26355	22915
29	29	26352	22888
30	30	25941	22481
31	31	26134	22[illegible]9
32	32	25837	22[illegible]53
33	33	25989	22[illegible]23
34	34	26351	22[illegible]93
35	35	26205	22[illegible]96
36	36	26616	22[illegible]48
37	37	26425	22[illegible]54
38	38	26583	23049

Ilustración 74: Archivo de estadísticas de calidad generado por Quality Statistics.

4.3.8 CALCULAR ESTADISTICAS DE COBERTURA

Este proceso es el encargado de generar un gráfico a partir de analizar el archivo BAM con los datos sobre la cobertura de las lecturas para cada posición del genoma, teniendo en cuenta las alineamientos únicos y múltiples.

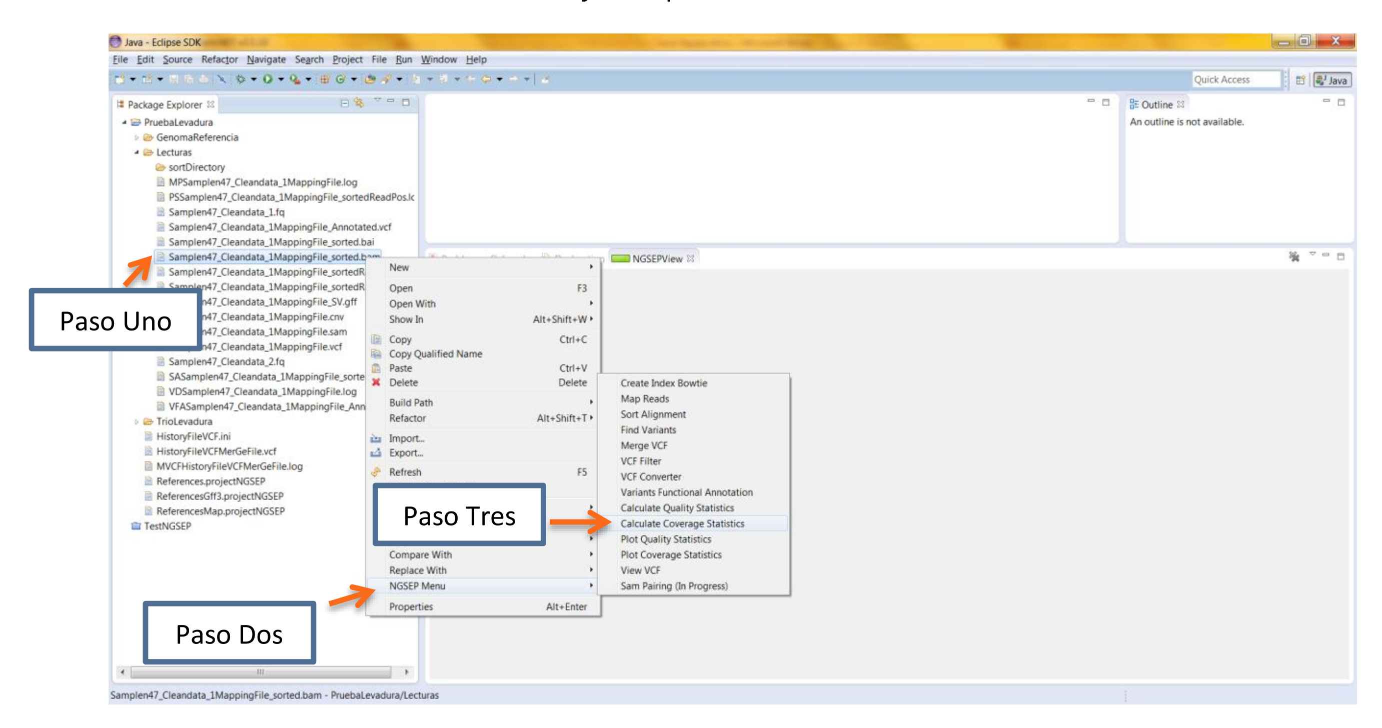


Ilustración 75: accediendo al proceso ''Calculated Coverage Statistics ''.

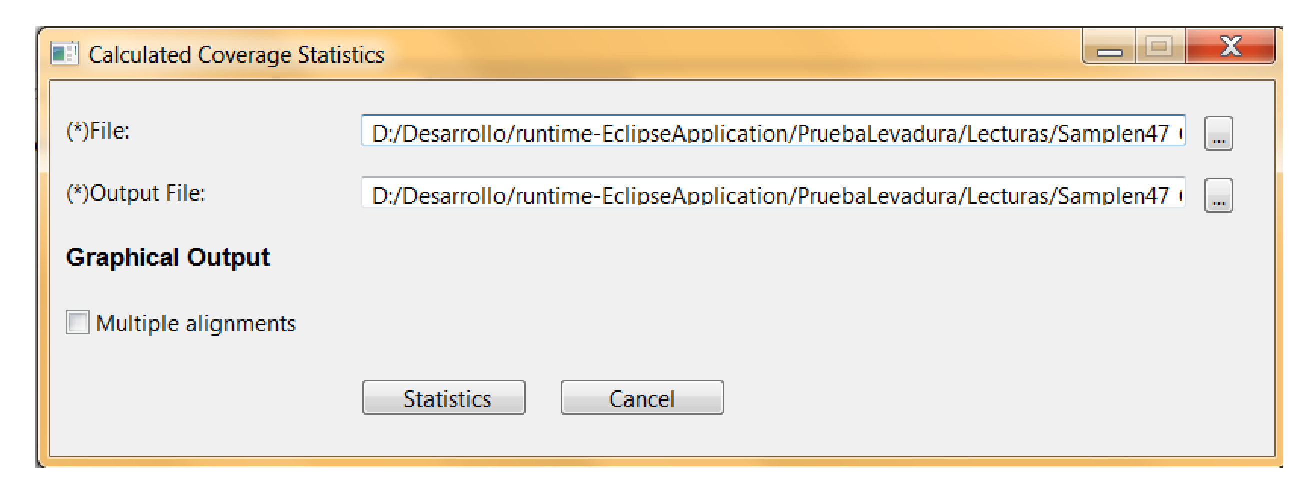


Ilustración 76: pantalla de ''Calculated Coverage Statistics ''.

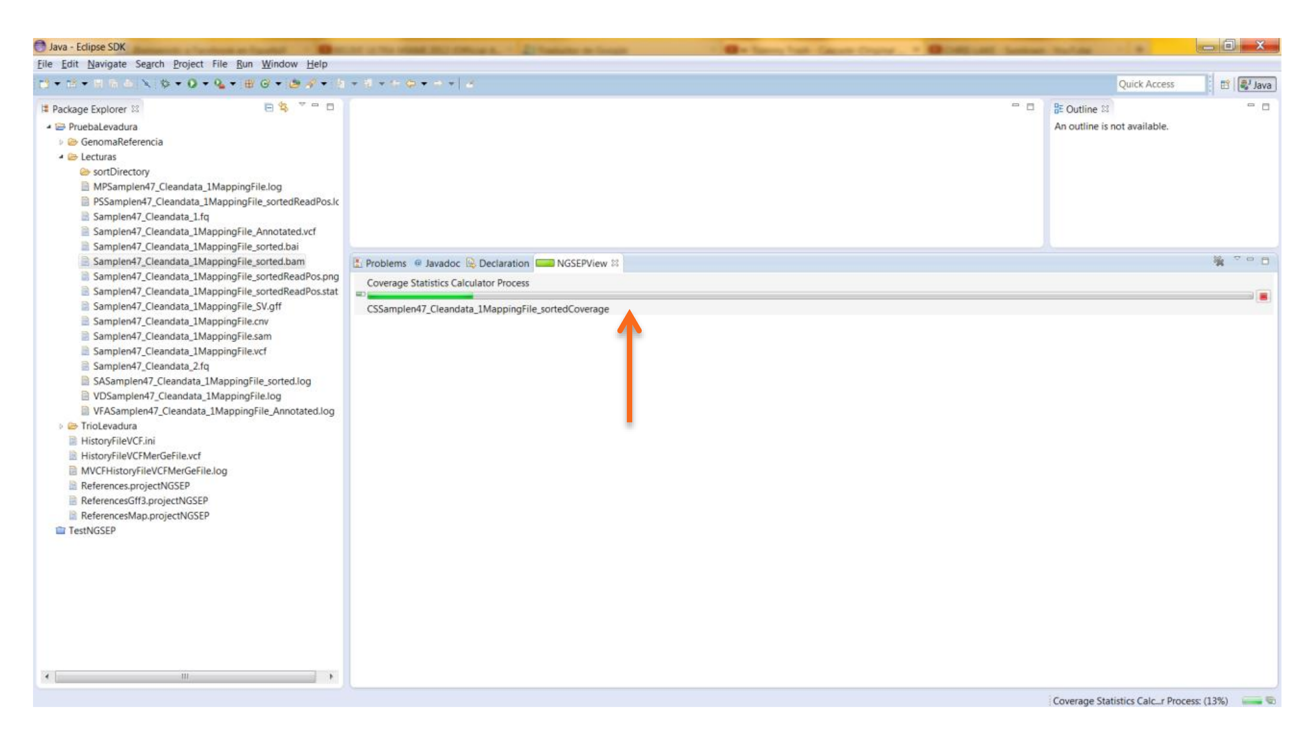


Ilustración 77: ejecución de ''calculate Coverage Statistics''.

Coverage Statistics corriendo, una vez finalizado este proceso se generar un archivo con las estadísticas sobre la cobertura de las lecturas para cada posición del genoma y la gráfica con respecto a los alineamientos únicos o múltiples de acuerdo a lo que el usuario ingrese.

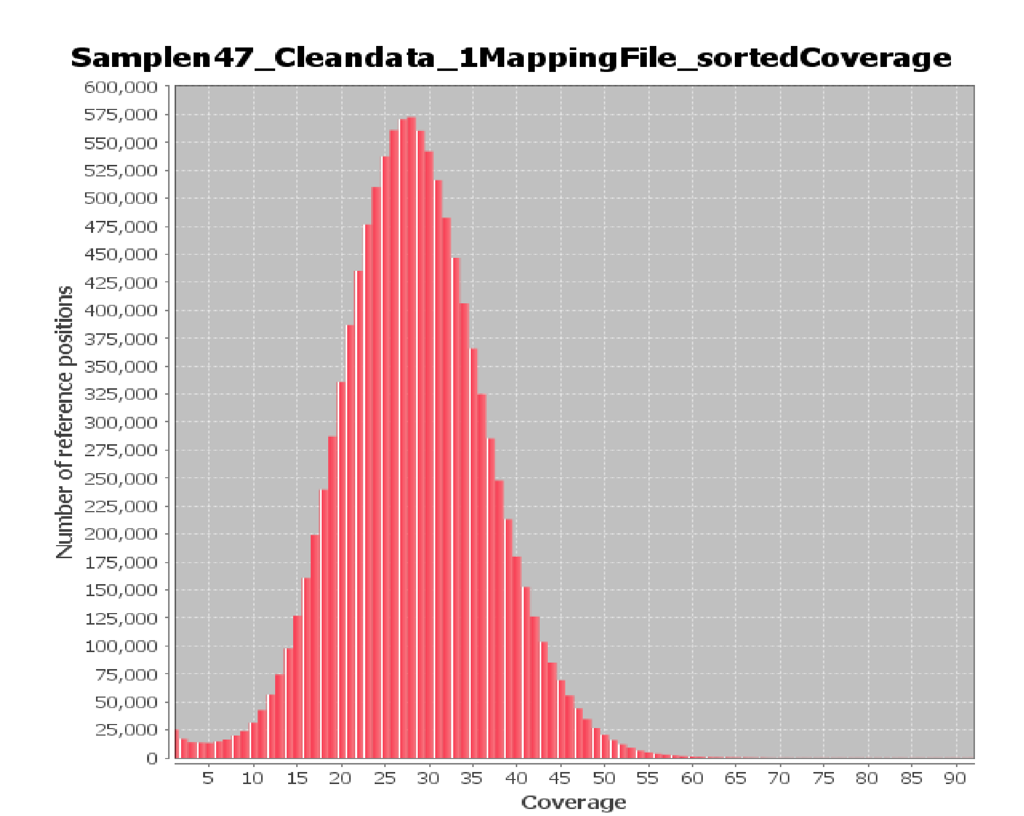


Ilustración 78: Grafica de cobertura.



Samplen47_Cleandata_1MappingFile_sortedCoverage.stats

1	1	59038	25394
2	2	44298	16987
3	3	37801	13861
4	4	32701	13541
5	5	29436	13350
6	6	28824	14709
7	7	27655	16351
8	8	30066	19816
9	9	32368	23858
10	10	38564	31306
11	11	47872	42813
12	12	60221	56564
13	13	78229	74354
14	14	100359	97731
15	15	129344	127089
16	16	163449	160806
17	17	202467	199242
18	18	244113	239704
19	19	293348	287432
20	20	342850	335856
21	21	394905	386635
22	22	444549	435464
23	23	485990	476710
24	24	520422	509954
25	25	550623	537520
26	26	574078	560809
27	27	584006	570558
28	28	586842	572526
29	29	574687	560449
30	30	558615	542032
31	31	531957	516162
32	32	498321	482808
33	33	462881	447084
34	34	421204	406181
35	35	379964	365995
36	36	338447	325091
37	37	298233	285452
38	38	259446	247996

Ilustración 79: archivo de estadísticas de cobertura.

Una vez realizados estos ocho procesos se estaría cumpliendo con orden establecido por el pipeline de NGSEP para detección de variantes genómicas.

4.4 COMPARATIVA DE NGSEP CONTRA LA HERRAMIENTA (SNVER) GANADORA DE LA EVALUACIÓN REALIZADA EN EL CAPÍTULO 2.

En este apartado, se realizará la comparativa entre la herramienta NGSEP expuesta en este capítulo, contra la herramienta ganadora, producto de la comparación realizada en el capítulo 2 pág. 52. Esta comparación se hace con el fin de conocer que tanta usabilidad tiene la (GUI) de NGSEP respecto a la herramienta número uno actualmente en implementación de usabilidad.

Teniendo en cuenta la escala de la Tabla 4 para calificar la usabilidad, definida en el capítulo 2 pág. 35, se procede a calificar la herramienta NGSEP bajo los mismos criterios de evaluación por los cuales fueron sometidos SNVER, GATK Y SAMTOOLS.

Las calificaciones para la evaluación, fueron otorgadas por un usuario con pleno conocimiento del contexto de herramientas bioinformáticas. Para una mayor apreciación de las calificaciones obtenidas por NGSEP, se elabora una muestra con imágenes del proceso de detección de variantes genómicas de NGSEP.

Las calificaciones obtenidas por SNVer son justificadas en la pág.38.

Herramienta (NGSEP)

Heurística: Visibilidad del estado del sistema

Pregunta: ¿La aplicación mantiene siempre informado al usuario del estado del sistema, así como de los caminos que este pueda tomar con una retroalimentación visual apropiada en tiempo razonable?

Una vez se dé clic derecho sobre el archivo BAM, automáticamente la aplicación se activara en el menú de opciones desplegadas por Eclipse, dentro de esas opciones se encuentra “NGSEP Menu”, dentro del menú de NGSEP se busca la opción “Find Variants” y se da clic sobre esta, inmediatamente se abre la pantalla del proceso Ilustración 80, luego de ingresados los datos correspondientes, automáticamente la aplicación crea una barra de progreso en la vista de procesos de NGSEP y muestra al usuario una retroalimentación visual de la ejecución del proceso de principio a fin de la misma Ilustración 81, también se genera un archivo log con información relevante del proceso lo cual mantiene informado al usuario del estado del sistema.

Variants Detector

(*)File : D:\Desarrollo\runtime-EclipseApplication\PruebaLevadura\prueba\Sample08_ye

(*)Reference File: D:\Desarrollo\runtime-EclipseApplication\TestNGSEP\Reference\sacCer_SGD_ref

(*)Output File Prefix: D:\Desarrollo\runtime-EclipseApplication\PruebaLevadura\prueba\Sample08_ye

Find Variants

Execution Parameters

Skip Repetitive Regions Detection

Skip New CNV Detection

Skip Structural Variants Detection

Skip SNVs Detection

CNVs Detection Parameters

Genome Size:

Bin Size: 100

SNVs Detection Parameters

Genomic Location:

Heterozygosity Rate: 0.001

Minimun Genotype Quality Score: 40

Maximun Base Quality Score: 30

Alternative Allele Coverage: Min: Max:

Ignore Lower Case Reference

Include Secondary Alignments

Genotype All Covered Sites

Maximun Alignment Per Start Position: 2

Ignore Bases 5': 0

Ignore Bases 3': 0

Known CNVs File:

Known Variants File:

Common Parameters

Ploidy: 2

(*)Sample Id: Sample08_yeast1

Find Variants Cancel

Ilustración 80: Pantalla de NGSEP para detectar SNPs e Indeles.

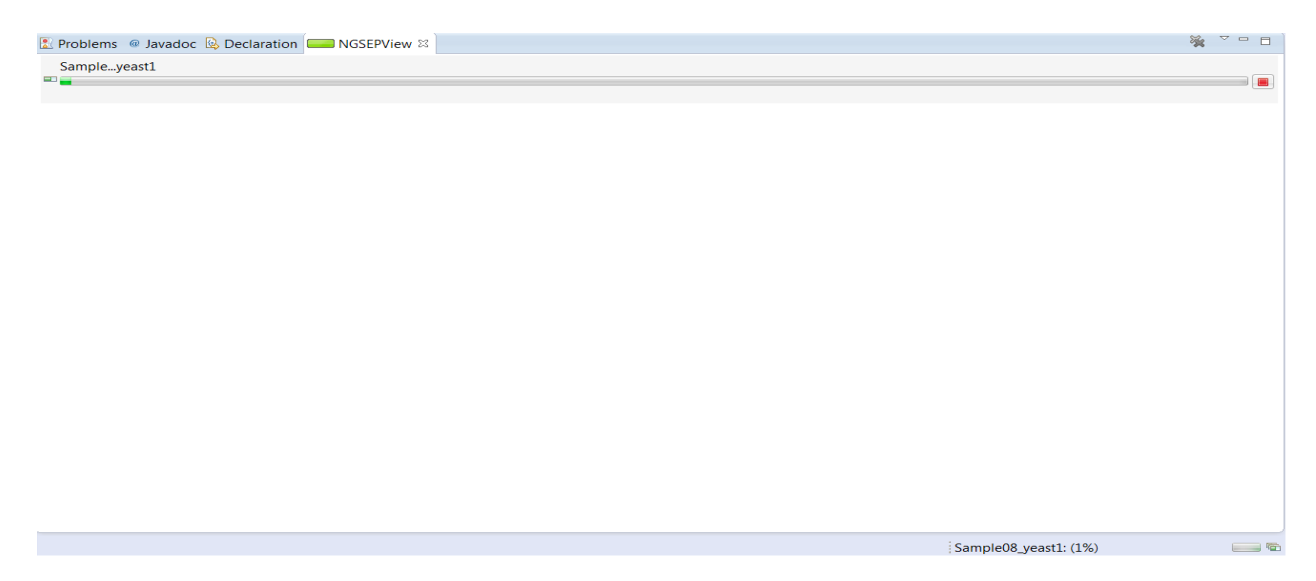


Ilustración 81: Barra de progreso del proceso “Find Variants” de NGSEP.

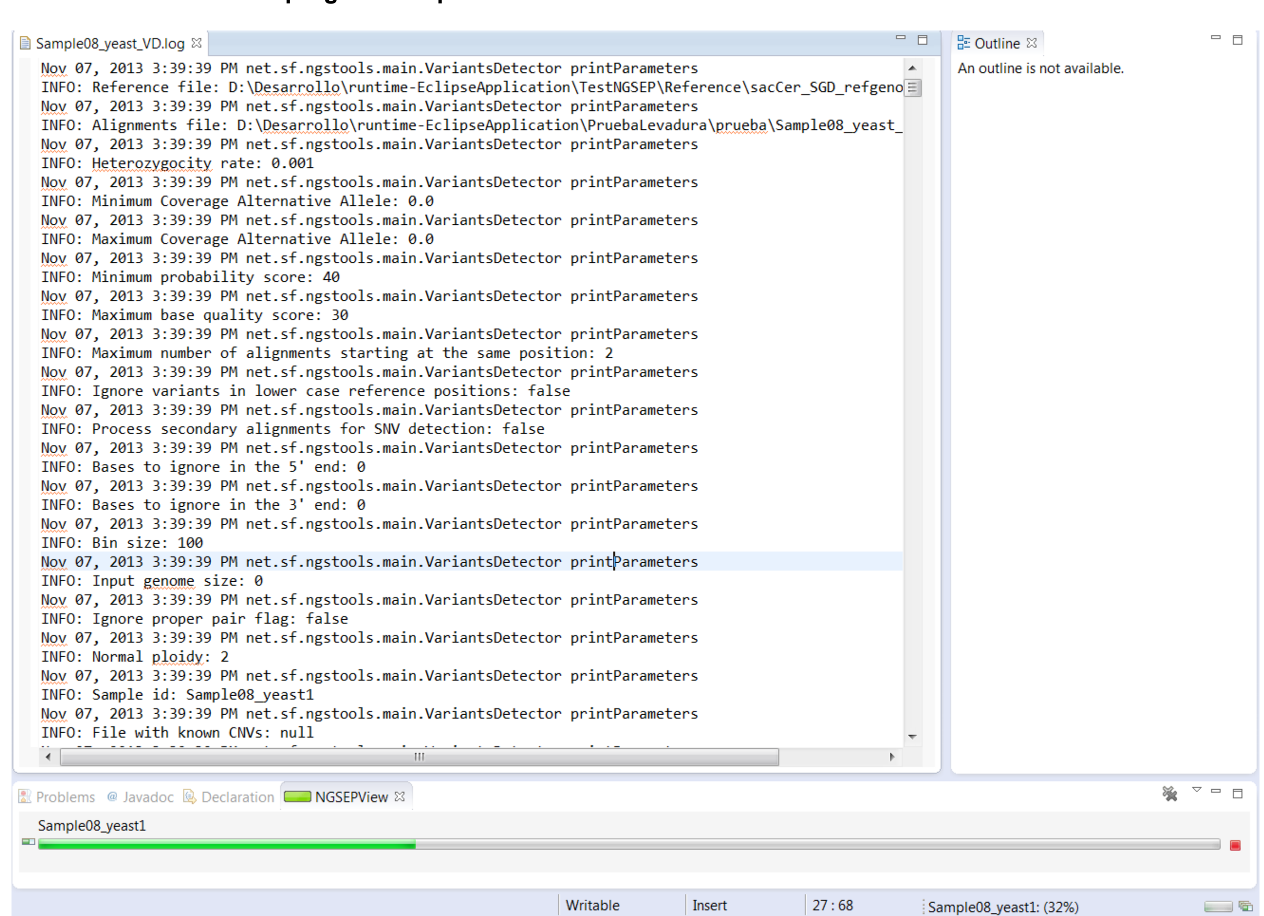


Ilustración 82: Log generado por el proceso de NGSEP con información relevante del proceso.

Calificación obtenida: la máxima calificación, cumple con la heurística de manera acorde a la pregunta realizada, calificación igual a 5.

Heurística: Control y libertad del usuario

Pregunta:

¿La interfaz de la aplicación permite controlar la iteración de los procesos, de esta manera dejando el control de la aplicación al usuario y permitiéndole interactuar con los elementos contenidos en la pantalla?

La interfaz gráfica de NGSEP permite controlar la iteración de cada proceso, mediante botones como los de correr el aplicativo o cancelar la ejecución Ilustración 84, Ilustración 85, también ofrece la posibilidad de visualizar al usuario una serie de pantallas pertenecientes a otros procesos una vez este corriendo el proceso actual Ilustración 83, permitiendo al usuario interactuar con los elementos que contiene las pantallas.

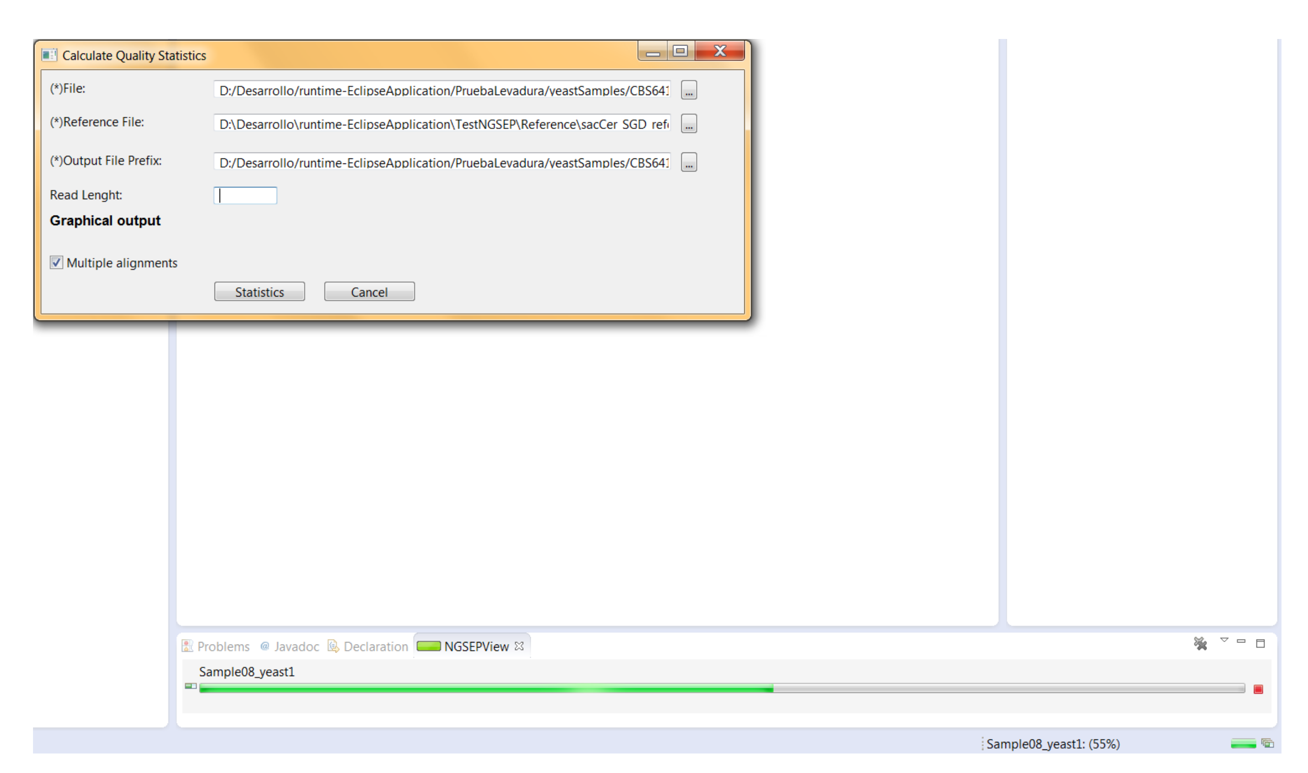


Ilustración 83: Pantalla del proceso calcular estadísticas de calidad de NGSEP abierto y en iteración con el usuario, mientras se ejecuta el proceso de detección de variantes de NGSEP.

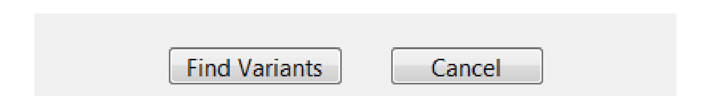


Ilustración 84: Botones para arrancar o cancelar la ejecución de detección de variantes proceso de NGSEP.

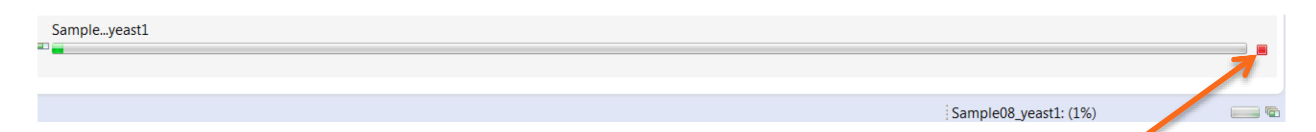


Ilustración 85: Botón para cancelar el proceso de detección de variantes de NGSEP en la pantalla.

Calificación obtenida: la máxima calificación, cumple con la heurística de manera acorde a la pregunta realizada, calificación igual a 5.

Heurística: Correspondencia entre el sistema y el mundo real

Pregunta: ¿La interfaz muestra mensajes en el idioma del usuario, cuando se habla de idioma se refiere a palabras, frases y conceptos familiares para el usuario, siempre en el contexto de la aplicación?

NGSEP maneja una cantidad de mensajes de diferente índole, tiene mensajes de error Ilustración 88, mensajes de ayuda al usuario Ilustración 86 y mensajes de información Ilustración 87.

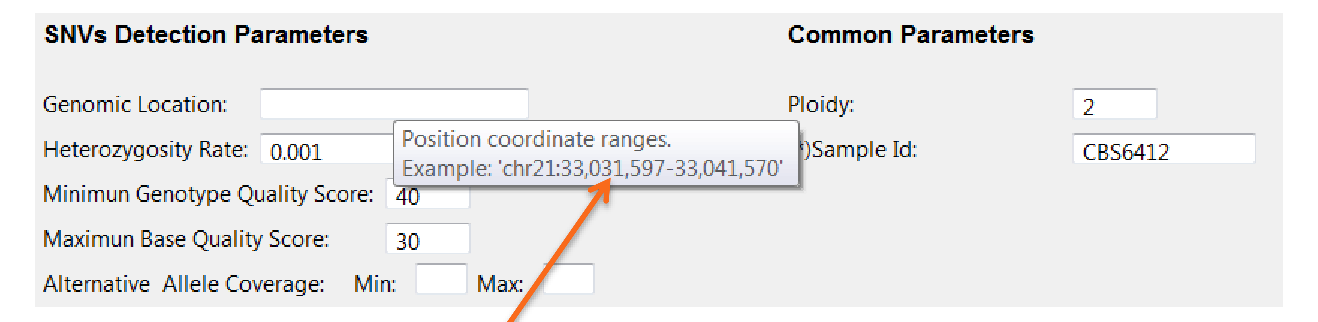


Ilustración 86: Mensaje de información para ayudar al usuario a digitar los datos en una entrada en el formato correcto, como muestra la sugerencia.

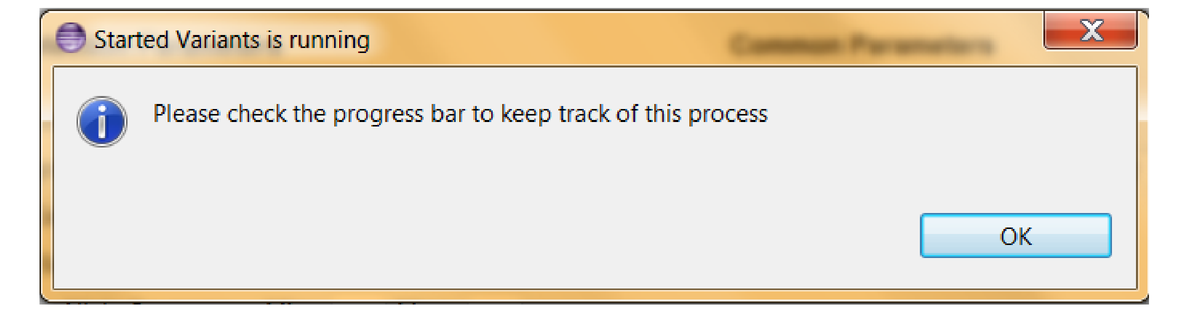


Ilustración 87: Mensaje para informar al usuario el comienzo de la ejecución del proceso.

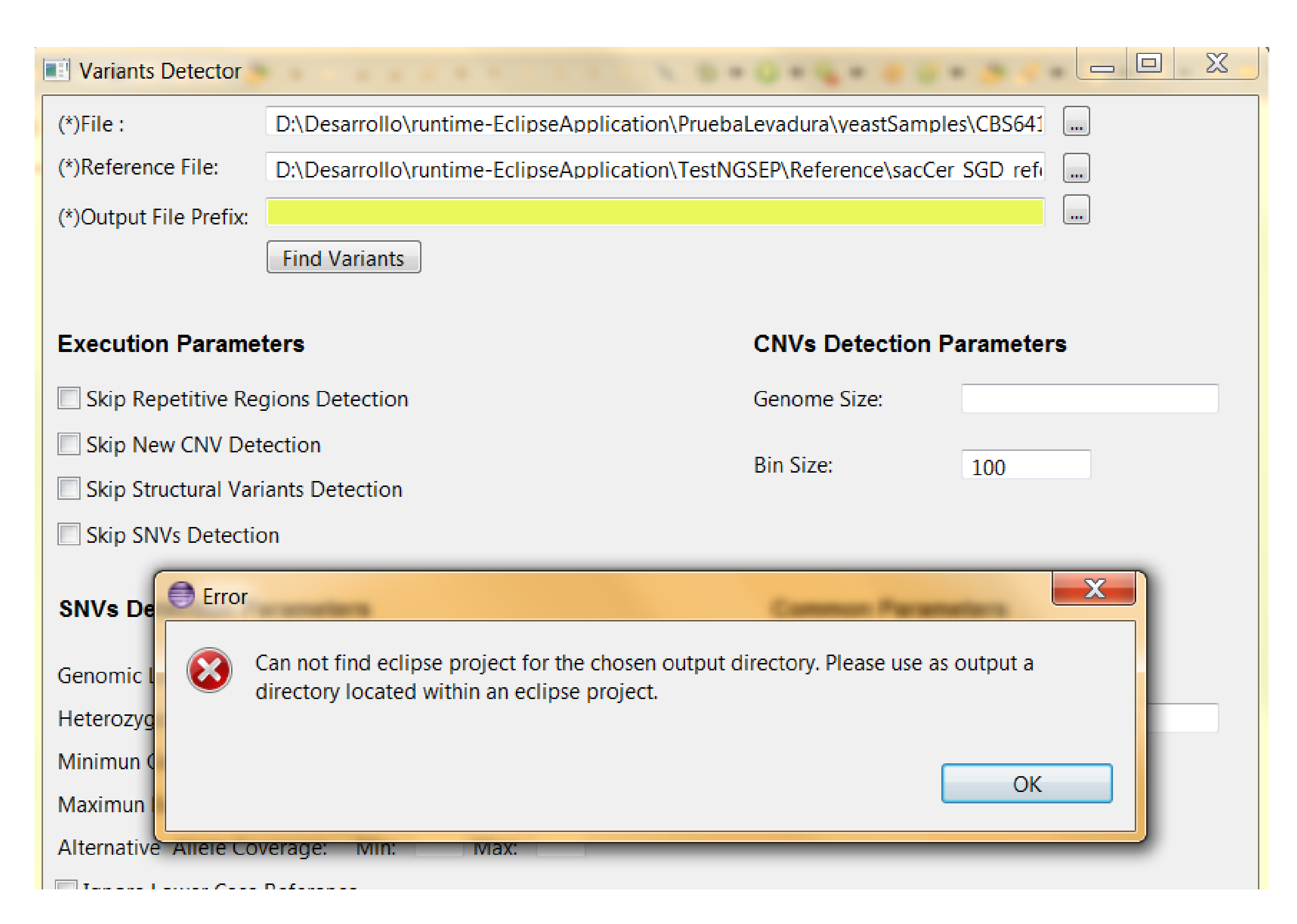


Ilustración 88: Mensaje de excepción al no ingresar un parámetro obligatorio para la ejecución del proceso.

Calificación obtenida: la máxima calificación, cumple con la heurística de manera acorde a la pregunta realizada, calificación igual a 5.

Heurística: Reconocer antes que recordar

Pregunta: ¿El diseño de la interfaz permite reducir la carga de memoria para un usuario final, se refiere a que si la interfaz ayuda al usuario a no tener que recordar información para ir de un proceso a otro a la hora de realizar una iteración?

NGSEP se compone de ocho procesos. En este sentido, NGSEP muestra de manera independiente las pantallas que pertenecen a los procesos Ilustración 90, pero las pantallas siguen estando en el mismo aplicativo; no se abren en otra parte por fuera de la aplicación como en SNVer, por otro lado el Menú de procesos de NGSEP está organizado de manera

secuencial Ilustración 89, permitiendo navegar de un proceso a otro, lo que genera que el usuario reconozca la información necesaria para comenzar el siguiente proceso al que actualmente ejecuta.

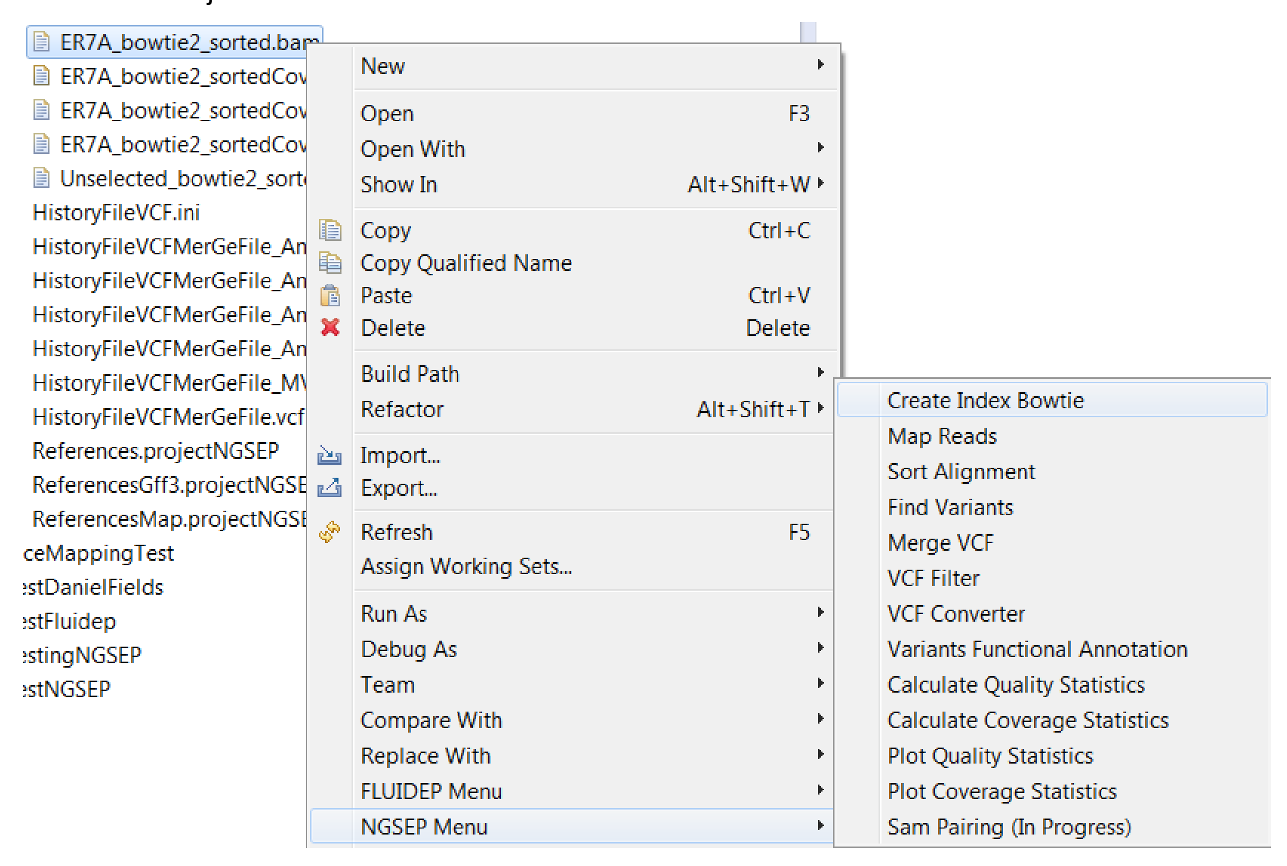


Ilustración 89: Menú de procesos de NGSEP organizado de manera que el usuario empiece el pipeline o flujo de trabajo de arriba hacia abajo.

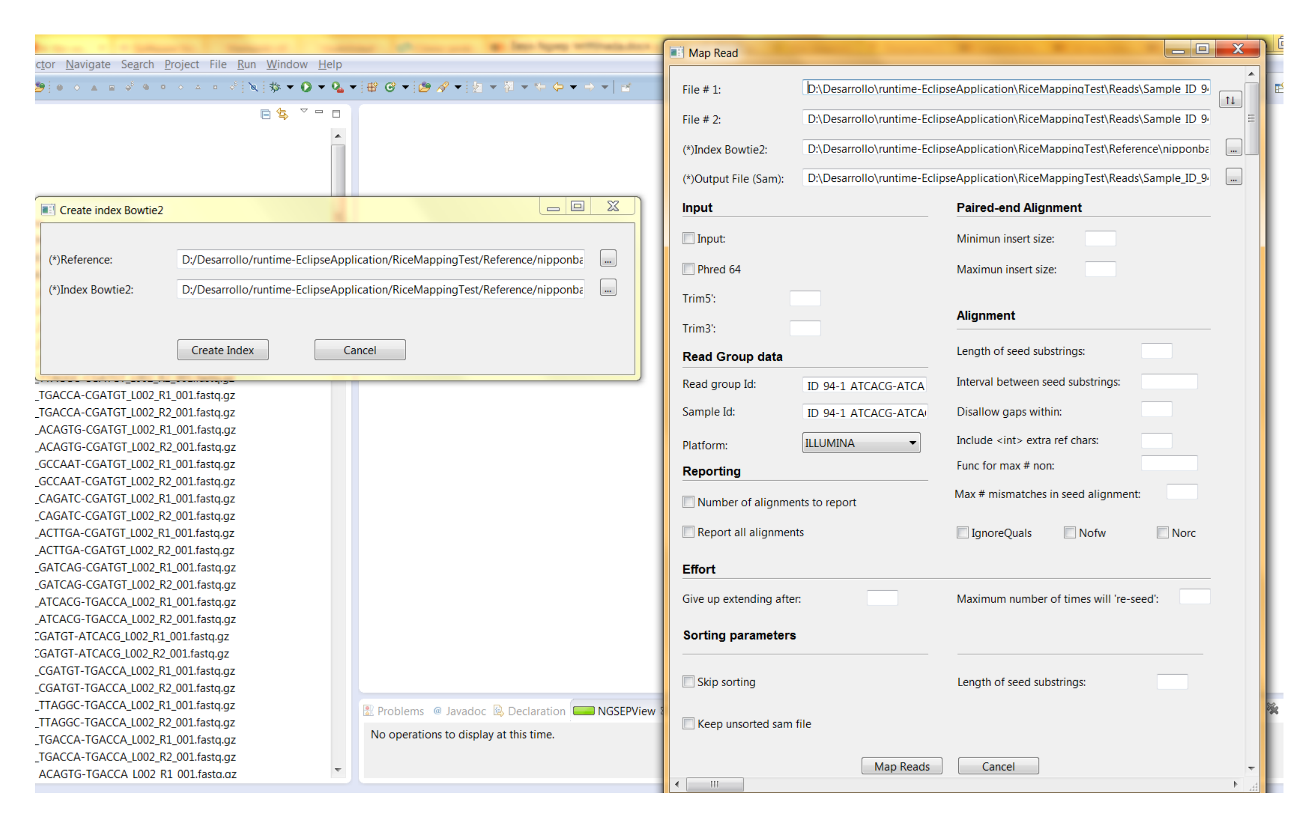


Ilustración 90: Dos procesos abiertos a la misma vez, el proceso de mapeo depende de la información generada por el primero crear índice de bowtie2.

Calificación obtenida: la máxima calificación, cumple con la heurística de manera acorde a la pregunta realizada, calificación igual a 5.

Heurística: Prevención de errores

Pregunta: ¿La Aplicación tiene un buen diseño de mensajes de error que den la posibilidad al usuario de retraerse antes de que se realice la acción y se comprometan los datos?

El manejo de mensajes de error de NGSEP da la posibilidad al usuario de retraerse antes de que se realice cualquier acción que genere fallos en los datos a generar Ilustración 91, Ilustración 92.

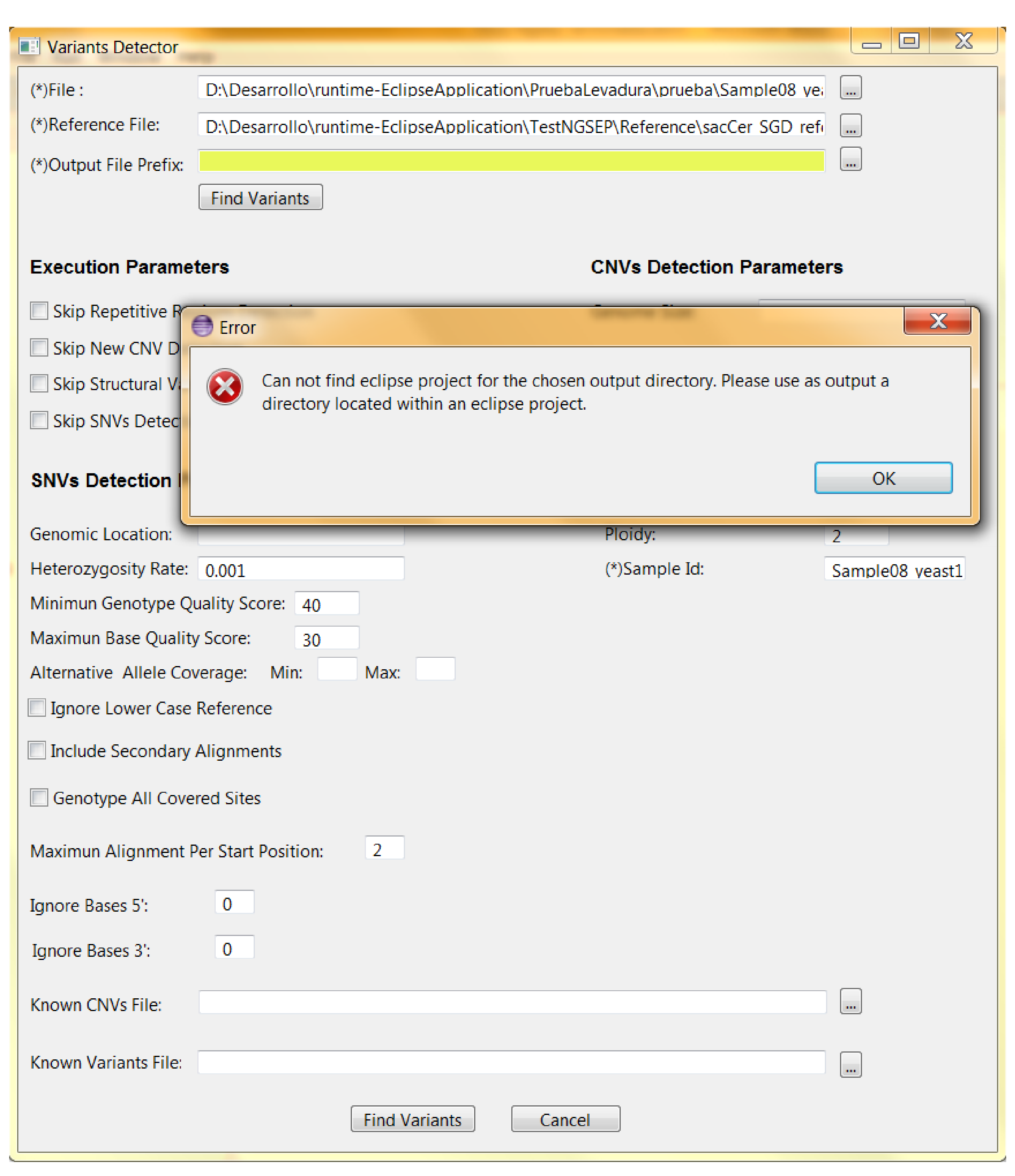


Ilustración 91: validaciones de campos y mensajes que advierten al usuario antes de ejecutar cualquier proceso.

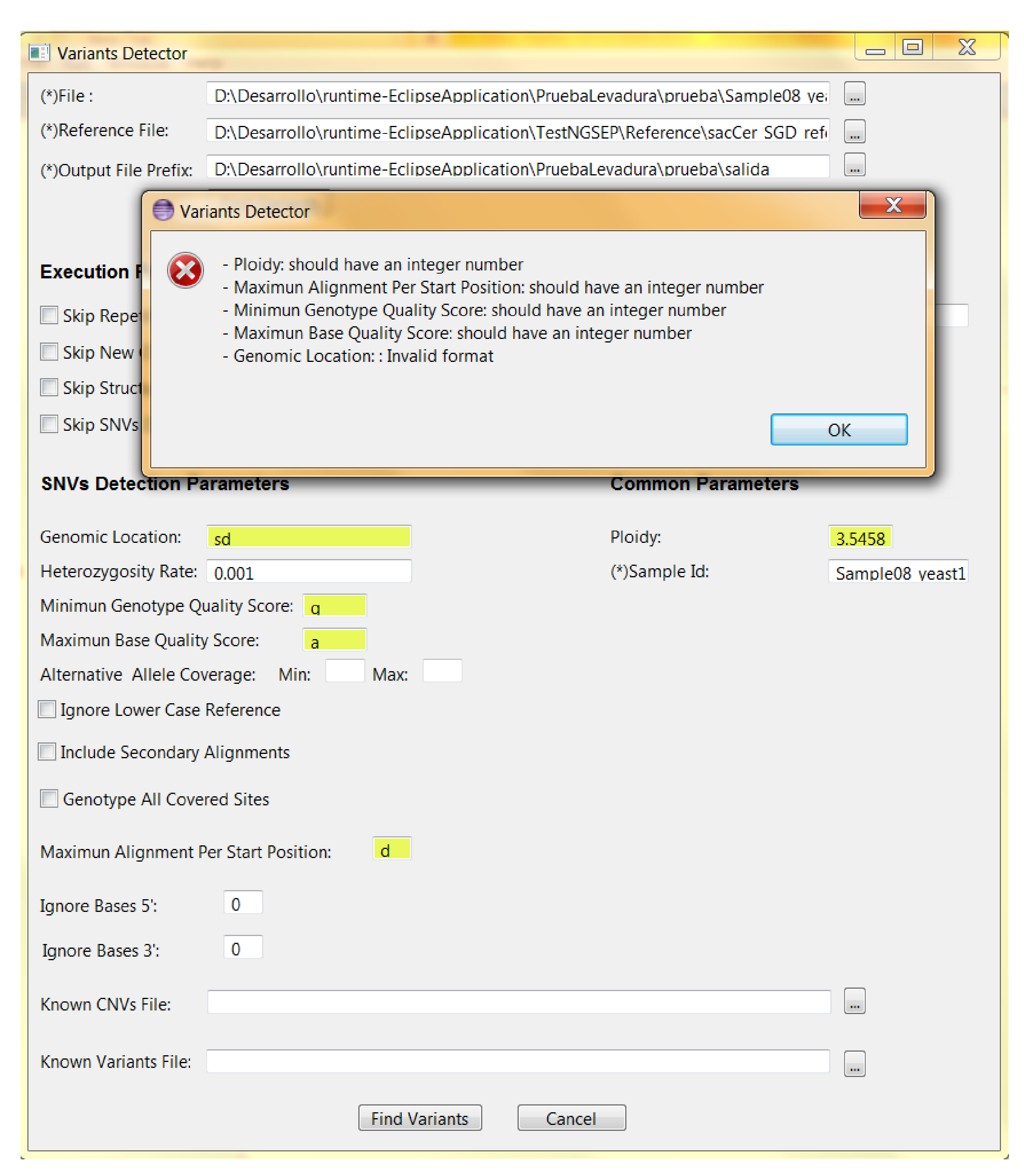


Ilustración 92: Errores en campos de la pantalla del proceso de detección de variantes.

Calificación obtenida: la máxima calificación, cumple con la heurística de manera acorde a la pregunta realizada, calificación igual a 5.

Heurística: Estética y diseño minimalista

Pregunta: ¿Los mensajes de la aplicación contienen información relevante para la tarea que está realizando el usuario, por otro lado el diseño de la interfaz es simple, fácil de aprender, fácil de usar y con fácil acceso a las funcionalidades que ofrece la aplicación?

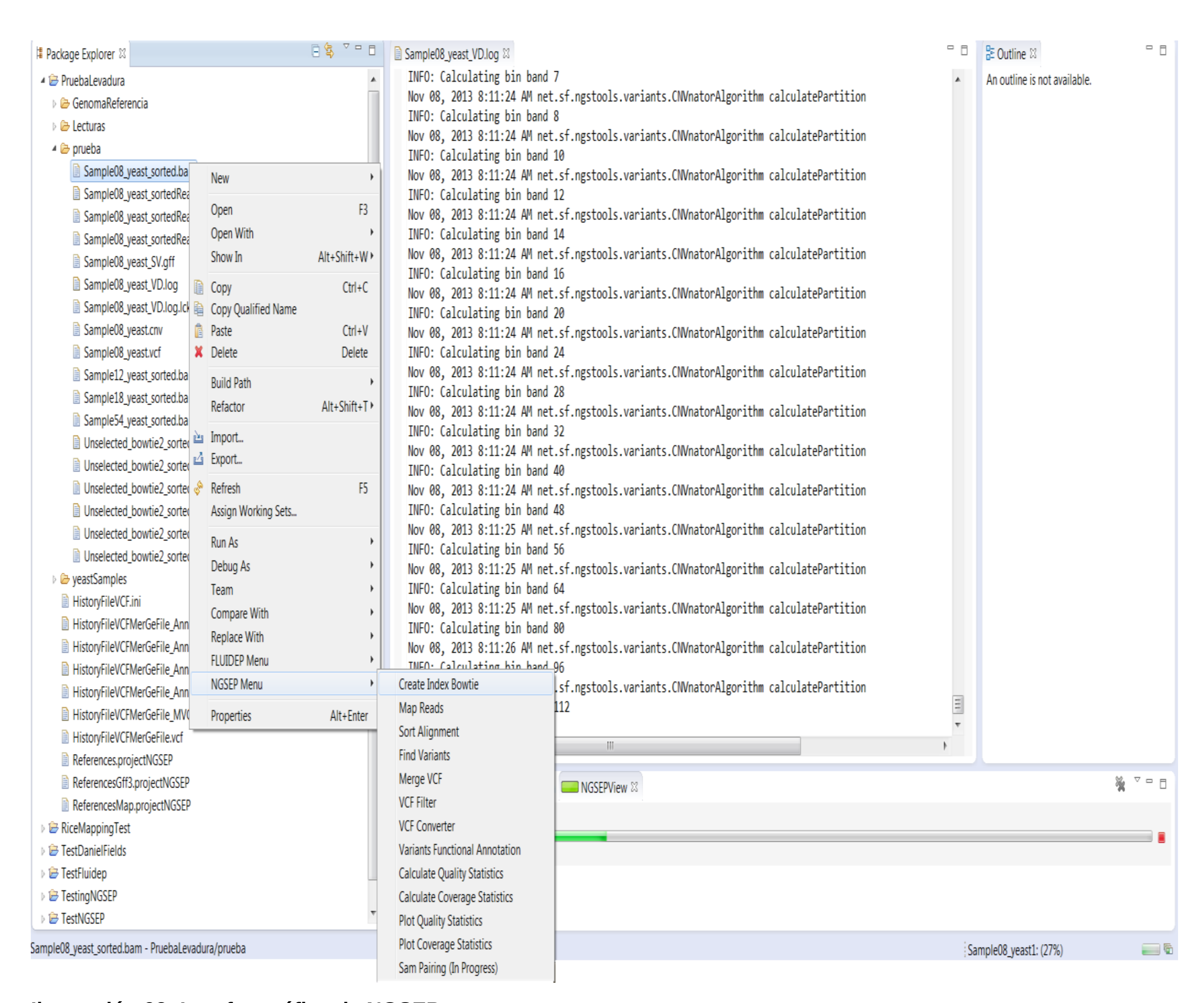


Ilustración 93: Interfaz gráfica de NGSEP.

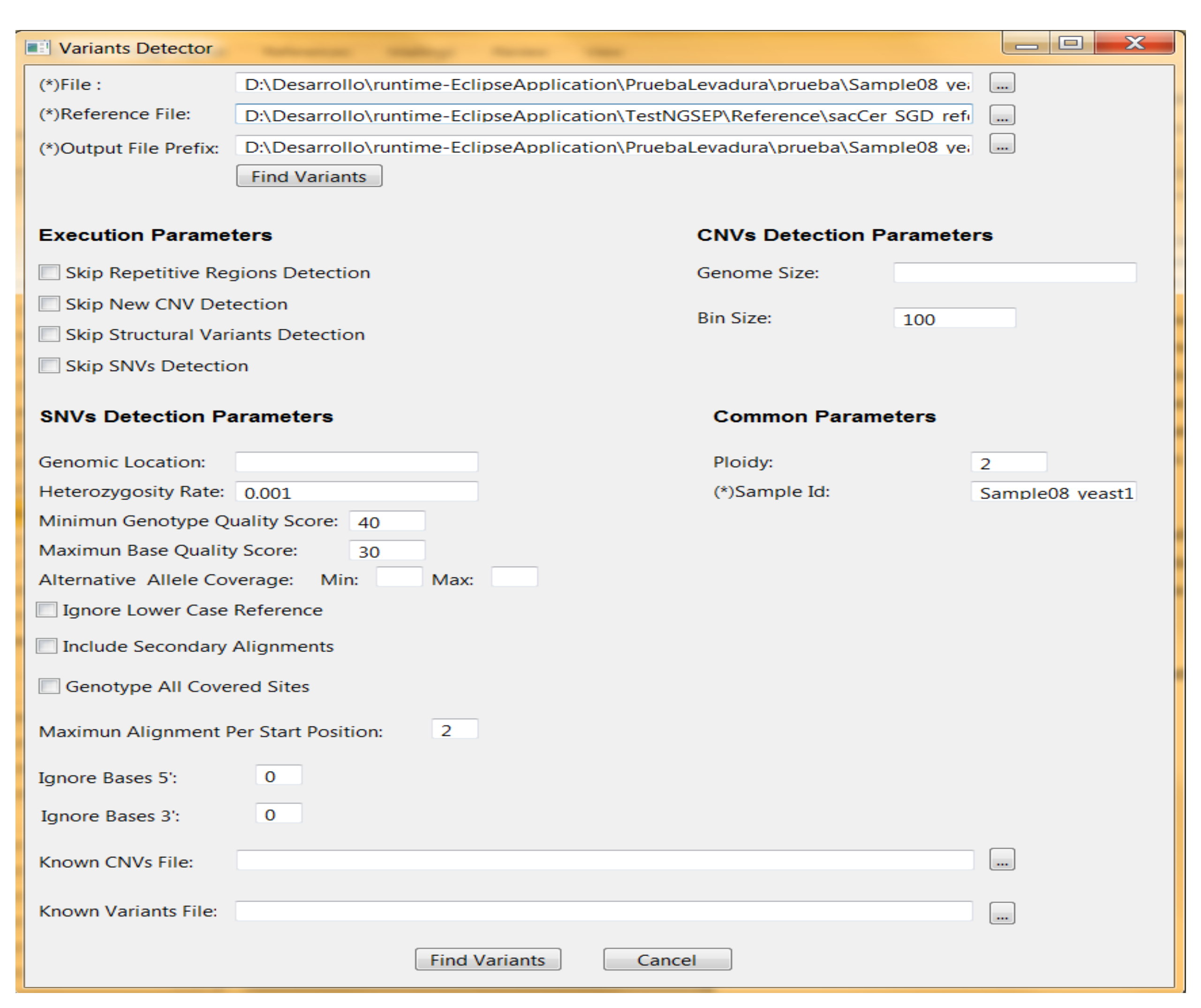


Ilustración 94: Interfaz gráfica del proceso de detección de variantes de NGSEP.

La Ilustración 94 muestra como la pantalla del proceso de detección de variantes de NGSEP tiene una interfaz simple ya que, tiene entradas con títulos de tamaños grandes y claros a la vista del usuario, además de contener botones acordes para arrancar el proceso y para cancelarlo, la aplicación en la Ilustración 93 muestra cómo se accede de manera fácil al menú del aplicativo y las funciones ofrecidas por NGSEP. La generación de mensajes con respecto a la información de los procesos es concisa y con información importante del estado actual del proceso Ilustración 82, Ilustración 87.

Calificación obtenida: la máxima calificación, cumple con la heurística de manera acorde a la pregunta realizada, calificación igual a 5.

Heurística: Ayudar a los usuarios a reconocer, diagnosticar y recuperar errores

Pregunta: ¿La aplicación tiene mensajes de error en lenguaje entendible por el usuario y sin código de lenguajes de programación, los mensajes indican el error y sugieren como solucionarlo?

NSGEP genera mensajes de error en un lenguaje entendible para el usuario final, también marca las caja de texto donde se genere el error por los datos ingresados del usuario, además de crear un dialogo de mensajes de error que indica que hay errores en diferentes entradas de la pantalla, además sugiere al usuario mediante los mensajes que tipo de dato debería de ingresar en un lenguaje entendible Ilustración 95.

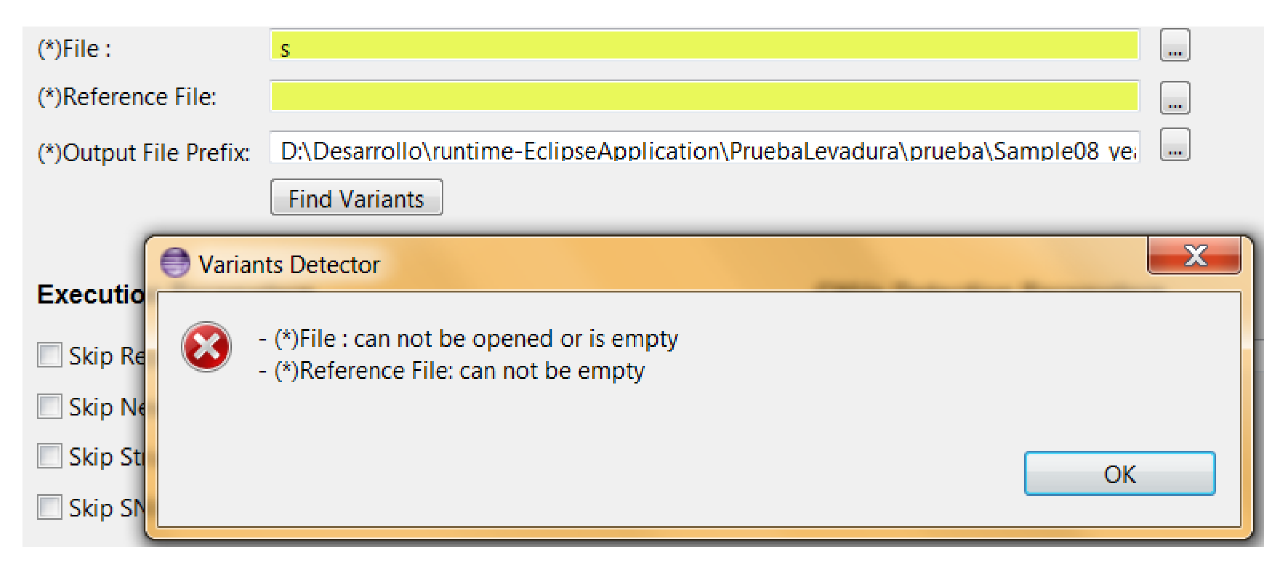


Ilustración 95: Mensaje de excepción de error en NGSEP.

Calificación obtenida: la máxima calificación, cumple con la heurística de manera acorde a la pregunta realizada, calificación igual a 5.

Heurística: Ayuda y documentación

Pregunta: ¿La aplicación tiene manual de usuario, la información es fácil de encontrar y enfocada a la tarea que el usuario realiza, se listan los pasos necesarios para la realización de la tarea?

NGSEP tiene un manual de usuario pág. 156 con contenido de fácil navegación, ofrece la posibilidad mediante links de ir al capítulo que el usuario desea, además explica en cada capítulo o sección de forma detalla para que sirve cada proceso y cada entrada que dato

recibe. También lista cada uno de los pasos necesarios para ejecutar los diferentes procesos de la aplicación.

Calificación obtenida: la máxima calificación, cumple con la heurística de manera acorde a la pregunta realizada, calificación igual a 5.

La Tabla 7 contiene las calificaciones otorgadas para las tres herramientas por un usuario con pleno conocimiento del contexto de herramientas bioinformáticas.

Heurística	Pregunta	NGSEP	SNVer
Visibilidad del estado del sistema	¿La aplicación mantiene siempre informado al usuario del estado del sistema, así como de los caminos que este pueda tomar con una retroalimentación visual apropiada en tiempo razonable?	**5**	**5**
Control y libertad del usuario	¿La interfaz de la aplicación permite controlar la iteración de los procesos, de esta manera dejando el control de la aplicación al usuario y permitiéndole interactuar con los elementos contenidos en la pantalla?	**5**	**5**
Correspondencia entre el sistema y el mundo real	¿La interfaz muestra mensajes en el idioma del usuario, cuando se habla de idioma se refiere a palabras, frases y conceptos familiares para el usuario, siempre en el contexto de la aplicación?	**5**	**5**
Reconocer antes que recordar	¿El diseño de la interfaz permite reducir la carga de memoria para un usuario final, se refiere a que si la interfaz ayuda al usuario a no tener que recordar información para ir de un proceso a otro a la hora de realizar una iteración?	**5**	**3**
Prevención de errores	¿La Aplicación tiene realizar un buen diseño de mensajes de error que den la posibilidad al usuario de retraerse antes de que se realice la acción y se comprometan los datos?	**5**	**5**
Estética y diseño minimalista	¿Los mensajes de la aplicación contienen información relevante para la tarea que está realizando el usuario, por otro lado el diseño de la interfaz es simple, fácil de aprender, fácil de usar y con fácil acceso a las funcionalidades que ofrece la aplicación?	**5**	**5**
Ayudar a los usuarios a reconocer, diagnosticar y recuperar errores	¿La aplicación tiene mensajes de error en lenguaje entendible por el usuario y sin código de lenguajes de programación, los mensajes indican el error y sugieren como solucionarlo?	**5**	**4**
Ayuda y documentación	¿La aplicación tiene manual de usuario, la información es fácil de encontrar y enfocada a la tarea que el usuario realiza, se listan los pasos necesarios para la realización de la tarea?	**5**	**5**

Tabla 7: Evaluación realizada con la escala de la Tabla 4Tabla 3 aplicada a las herramientas NGSEP y SNVer.

Heurística	Pregunta	NGSEP	SNVer
Visibilidad del estado del sistema	¿La aplicación mantiene siempre informado al usuario del estado del sistema, así como de los caminos que este pueda tomar con una retroalimentación visual apropiada en tiempo razonable?	**12.5**	**12.5**
Control y libertad del usuario	¿La interfaz de la aplicación permite controlar la iteración de los procesos, de esta manera dejando el control de la aplicación al usuario y permitiéndole interactuar con los elementos contenidos en la pantalla?	**12.5**	**12.5**
Correspondencia entre el sistema y el mundo real	¿La interfaz muestra mensajes en el idioma del usuario, cuando se habla de idioma se refiere a palabras, frases y conceptos familiares para el usuario, siempre en el contexto de la aplicación?	**12.5**	**12.5**
Reconocer antes que recordar	¿El diseño de la interfaz permite reducir la carga de memoria para un usuario final, se refiere a que si la interfaz ayuda al usuario a no tener que recordar información para ir de un proceso a otro a la hora de realizar una iteración?	**12.5**	**7.5**
Prevención de errores	¿La Aplicación tiene realizar un buen diseño de mensajes de error que den la posibilidad al usuario de retraerse antes de que se realice la acción y se comprometan los datos?	**12.5**	**12.5**
Estética y diseño minimalista	¿Los mensajes de la aplicación contienen información relevante para la tarea que está realizando el usuario, por otro lado el diseño de la interfaz es simple, fácil de aprender, fácil de usar y con fácil acceso a las funcionalidades que ofrece la aplicación?	**12.5**	**12.5**
Ayudar a los usuarios a reconocer, diagnosticar y recuperar errores	¿La aplicación tiene mensajes de error en lenguaje entendible por el usuario y sin código de lenguajes de programación, los mensajes indican el error y sugieren como solucionarlo?	**12.5**	**10**
Ayuda y documentación	¿La aplicación tiene manual de usuario, la información es fácil de encontrar y enfocada a la tarea que el usuario realiza, se listan los pasos necesarios para la realización de la tarea?	**12.5**	**12.5**
Total		**100**	**92.5**

Tabla 8: resultados de la evaluación realizada en la Tabla 7.

4.5 GRAFICA COMPARATIVA DE LA Tabla 8.

De acuerdo, a lo definido en el apartado GRÁFICA COMPARATIVA DE LA Tabla 6. de la pág. 51 del capítulo 2, se procede a graficar los valores obtenidos por las herramientas NGSEP Y SNVER en la Tabla 8 con respecto a las heurísticas establecidas en el apartado de la pág. 35 en el capítulo 2.

Ilustración 96, expresa de manera gráfica las distancias entre los valores reales. Permite evaluar el desempeño de una herramienta respecto a las heurísticas establecidas en el apartado de la pág. 35 en el capítulo 2.

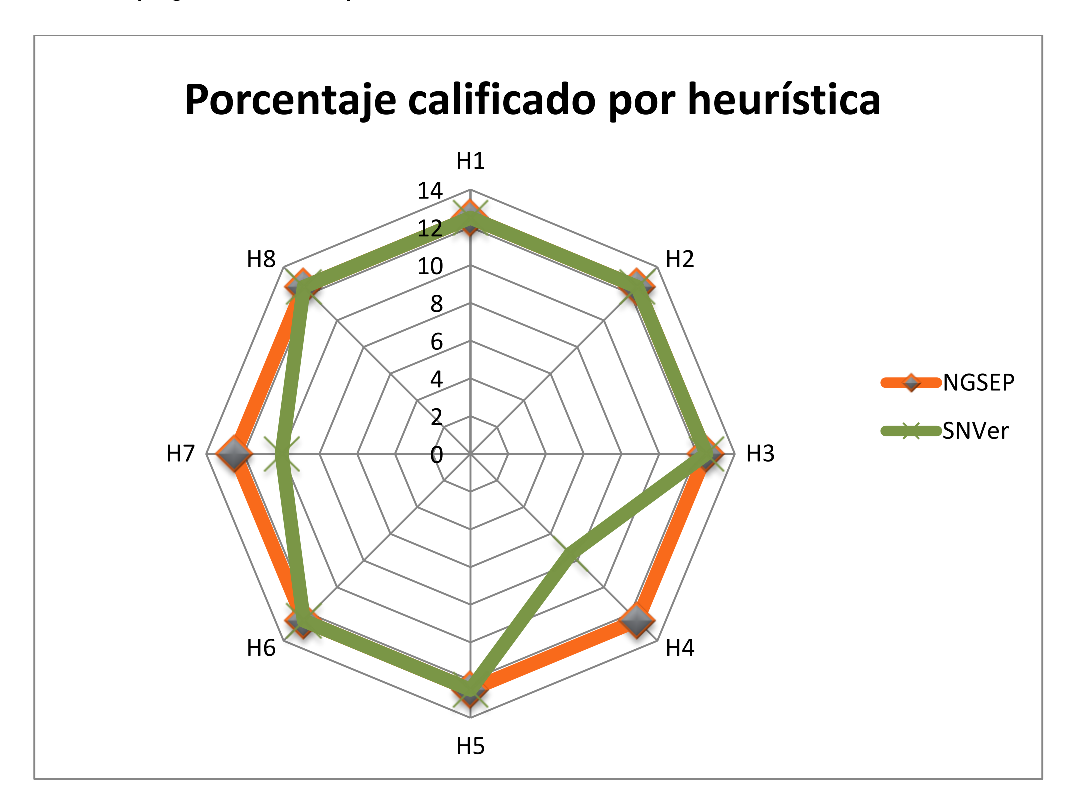


Ilustración 96: Grafica producto de los valores obtenidos por las herramientas evaluadas respecto a las 8 heurísticas de usabilidad.

De acuerdo a los resultados presentados en la Ilustración 96, se puede observar que NGSEP es superior a SNVER respecto a la heurística 4 (Reconocer antes que recordar) y en la heurística 7 (Ayudar a los usuarios a reconocer, diagnosticar y recuperar errores), en las demás heurísticas presentan similar porcentaje de eficacia.

4.6 GRÁFICA TOTAL DE USABILIDAD

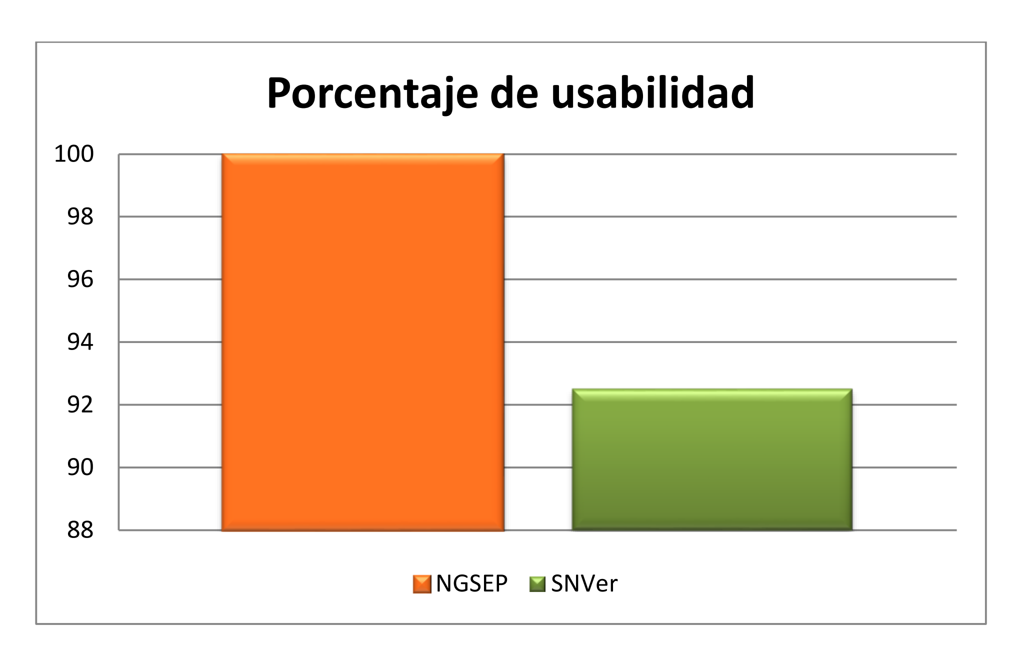


Ilustración 97: Grafica producto del porcentaje total obtenido por cada las herramientas evaluadas respecto a las 8 heurísticas de usabilidad.

A partir de los resultados presentados en la Ilustración 97, se concluye que el porcentaje obtenido por NGSEP respecto SNVER es mayor, en las heurísticas "Ayudar a los usuarios a reconocer, diagnosticar y recuperar errores" y "Reconocer antes que recordar" estos aspectos hace de NGSEP la mejor elección para un usuario final a la hora de trabajar con datos NGS, puesto que ofrece mayor usabilidad que SNVer en un aspecto tan importante como ayudar al usuario final a recuperar los errores producidos en cualquier proceso. NGSEP proporciona al usuario una serie de interfaces muy intuitivas, fáciles de entender y que permiten al usuario mantener el control de la ejecución de cada uno de los procesos contenidos en el menú de NGSEP.

CAPÍTULO 5: CONCLUSIONES Y TRABAJOS FUTUROS

5.1 CONCLUSIONES

- ✓ NGSEP representa una opción importante para científicos y demás usuarios con pocos conocimientos de programación y que desean utilizar una herramienta fácil de entender, para trabajar con datos de secuenciación de alto rendimiento.

- ✓ Las interfaces gráficas de NGSEP permiten fácilmente sugerirle al usuario parámetros por defecto en caso de que este no conozca los conceptos que se relacionan dentro de la herramienta.

- ✓ Utilizar interfaces gráficas para NGSEP hace posible dividir por pantallas los diferentes procesos contenidos dentro del pipeline de NGSTools. Esta división permite al usuario final tener un orden a la hora de ejecutar el pipeline.

- ✓ El uso de las heurísticas propuestas por Jakob Nielsen para evaluar la usabilidad en el diseño de interfaz gráfica de usuario permite medir que tan fácil es de usar NGSEP para un usuario final, de igual forma permitió realizar una comparativa de NGSEP con respecto a otras herramientas con un flujo de trabajo similar, obteniendo resultados positivos que justifican por que NGSEP está en un nivel superior a otras herramientas como: GATK, SNVer, SAMTools respecto a usabilidad.

- ✓ NGSEP va permitir a los científicos acelerar sus investigaciones en la mejora de cultivos, gracias al manejo de datos de tecnología de secuenciación de alto rendimiento, esto se debe a que NGSEP integra herramientas como bowtie2, Picard y BreakDancer antiguamente separadas y poco fáciles de manejar para los investigadores; en este sentido, NGSEP ofrece una solución completa que garantiza un flujo de trabajo secuencial.

- ✓ La integración de NGSEP a Eclipse garantiza que ofrezca un excelente sistema de organización de archivos, permitiendo de esta manera mantener un orden a la hora generar información a partir de datos NGS.

- ✓ Al integrar NGSEP a Eclipse se garantiza que proporciona un entorno multiplataforma, fácil de usar y extensible para el análisis de datos de NGS.

- ✓ La integración de NGSEP a Eclipse permite monitorear de principio a fin la ejecución de cada uno de los procesos contenidos dentro del menú de NGSEP.

5.2 TRABAJOS FUTUROS

- ✓ Mejorar los algoritmos de detección de CNV y SNVs.

- ✓ Desarrollar la opción de sincronizar automáticamente el editor de Plug-ins de Eclipse con las nuevas actualizaciones que se generen en NGSEP, sustituyendo la necesidad de descargar y volver a instalar cada vez que se genere una versión nueva, sino que Eclipse automáticamente descargue las actualizaciones y las instale.

- ✓ Actualmente, se está construyendo la opción de realizar múltiples mapeos con un solo clic a una carpeta contenedora de lecturas de un mismo organismo, de esta manera se estaría realizando un multi mapeo en paralelo ahorrando muchísimo tiempo.

- ✓ De igual forma, se está construyendo la opción de realizar detección de variantes genómicas a muchas muestras con tan solo un clic.

- ✓ Actualmente, se construye un visor de archivos vcf que permita conocer el estado actual de un vcf que se esté generando un proceso del Plug-in.

REFERENCIAS BIBLIOGRÁFICAS

1. FRANKHAM, R., BALLOU, J., BRISCOE, D. (2002). Introduction to Conservation Genetics. Cambridge University Press, united kingdom.

2. RAMANATHA, R.V., HODGKING, T. (2002). Genetic diversity and conservation and utilization of plant genetic resources. Plant Cell, Tissue and Organ Culture 68: 1-19.

3. BRACK A. (2000). Diversidad biológica y mercados en Perú: el problema agrario en debate. SEPIA VIII. Lima, Perú, 443-501.

4. FAO, CSFD, IPGRI. (2002). Conservación y ordenación de recursos genéticos forestales: en bosques naturales ordenados y áreas protegidas (in situ). Instituto Internacional de Recursos Genéticos.

5. DAVID T. SUZUKI. (2002) Genética. MCGRAW-HILL / INTERAMERICANA DE ESPAÑA, S.A., 2002. 7ª Ed. Págs. 67-99.

6. VALLEJO A. ESTRADA E. (2002). Mejoramiento genético de plantas. Palmira, Colombia: Universidad Nacional de Colombia. P.65-70.

7. DUITAMA, J., SRIVASTAVA, P. K., AND MANDOIU, I. I. (2012). Towards accurate detection and genotyping of expressed variants from whole transcriptome sequencing data. BMC Genomics, 13(Suppl 2), S6.

8. NIELSEN J (1993). Usability Engineering. Elsevier Science, ISBN-13: 9780125184069.

9. SANGER F, COULSON AR (1975). A rapid method for determining sequences in DNA by primed synthesis with DNA polymerase. J. Mol. Biol. 94 (3): 441–448.

10. ILLUMINA, INC (2011-2013). An Introduction to Next-Generation Sequencing Technology. (Disponible en: http://www.illumina.com/Documents/products/Illumina_Sequencing_Introduction.pdf. Consultado el: 20 de marzo del 2013).

11. BIOO SCIENTIFIC CORP, (2013). NGS Spotlight – Single Nucleotide Polymorphisms. (Disponible en: http://blog.biooscientific.com/ngs-spotlight-single-nucleotide-polymorphisms/. Consultado el: 20 de agosto del 2013).

12. JOSE A. PEREZ, (2004). Mutación ADN. (disponible en: http://press2.nci.nih.gov/sciencebehind/cancersp/cancersp42.htm. Consultado el: 24 de Mayo de 2013).

13. BROAD INSTITUTE, (2012). GATK. (Disponible en: http://www.broadinstitute.org/gatk/index.php. Consultado el: 15 de Julio del 2013).

14. LI H., HANDSAKER B., WYSOKER A., FENNELL T., RUAN J., HOMER N., MARTH G., ABECASIS G., Durbin R. and 1000 Genome Project Data Processing Subgroup (2009) The Sequence alignment/map (SAM) format and SAMtools. Bioinformatics, 25, 2078-9. [PMID: 19505943].

15. WEI Z, WANG W, HU P, LYON GJ AND HAKONARSON H. SNVer: a statistical tool for variant calling in analysis of pooled or individual next-generation sequencing data, Nucleic Acids Research (2011), [PMID: 21813454].

16. WATSON, JAMES D. AND FRANCIS H.C. CRICK, (1953). a structure for deoxyribose nucleic acid. *nature* 171, 737–738.

17. ALBERTS B, JOHNSON A, LEWIS J, RAFF M, ROBERTS K & WLATER P (2002). Molecular Biology of the Cell (4th ed.). Garland Science. ISBN 0-8153-3218-1. pp. 120-121.

18. ANSORGE, W.J. (2009). Next-generation DNA sequencing techniques.

19. M. GONZALO CLAROS, (2006). Vocabulario inglés-español de bioquímica y biología molecular. (Disponible en: http://www.biorom.uma.es/contenido/Glosario/. Consultado el: 30 de Julio de 2013).

20. NCBI. (2007). Resequencing. (Disponible en: http://www.ncbi.nlm.nih.gov/genome/probe/doc/TechResequencing.shtml. Consultado el: 23 de marzo del 2013).

21. PETER J. A. COCK, CHRISTOPHER J. FIELDS, NAOHISA GOTO, MICHAEL L. HEUER, PETER M. Rice Nucleic Acids Res. 2010 April; 38(6): 1767–1771. Published online 2009 December 16. PMCID: PMC2847217.

22. WANG L, JIANG T. (1994). On the complexity of multiple sequence alignment. J Comput Biol 1:337-348.

23. JUST W. (2001). Computational complexity of multiple sequence alignment with SP-score. J Comput Biol 8(6):615-23.

24. 1000 GENOMES. (2013). VCF (Variant Call Format) version4.1. (Disponible en: http://www.1000genomes.org/wiki/Analysis/Variant%20Call%20Format/vcf-variant-call-format-version-41. Consultado el: 30 de junio del 2013).

25. INFORMATICASONS. (2011). (Disponible en: http: //www.sosinformatica.net/evi/VisualBasic/guia_rapida/vb_guia_bd01.htm, Consultado el: 24 de mayo del 2013).

26. ECLIPSE. (2013). (Disponible en: http://www.eclipse.org/documentation/. Consultado el: 22 de Marzo del 2013).

27. FONTANILLO L, GONZÁLEZ R. (2005). proyecto eclipse. (Disponible en: http://zarza.usal.es/~fgarcia/docencia/poo/04-05/Trabajos/Eclipse.pdf. Consultado el: 22 Marzo del 2013).

28. ECLIPSE. (2013). SWT: The Standard Widget Toolkit. (Disponible en: http://www.eclipse.or http://www.eclipse.org/swt/g/swt/. Consultado el: 22 de Marzo del 2013).

29. ECLIPSE. (2013). JFace (Disponible en: http://wiki.eclipse.org/JFace/. Consultado el: 22 de Marzo del 2013).

30. SANTOS J, (2011). EPIDEMIOLOGÍA genética. (Disponible en: http://contacto.med.puc.cl/interconsulta/intercon_marzo_2011/Indice-Libro.pdf. Consultado el: 25 de junio del 2013).

31. NCBI. (2007). FASTA. (Disponible en: http://es.wikipedia.org/wiki/Formato_FASTA. Consultado el: 25 de junio del 2013).

32. QIU, F., Y. XU, K. LI, Z. LI, Y. LIU, H. DUANMU, S. ZHANG, Z. LI, Z. CHANG, Y. ZHOU, R. ZHANG, S. ZHANG, C. LI, Y. ZHANG, M. LIU AND X. LI (2012). CNVD: Text mining-based copy number variation in disease database. Hum Mutat 33(11): E2375-2381. PubMed ID: 22826268.

33. MOLICH, R., AND NIELSEN, J. (1990). Improving a human-computer dialogue, Communications of the ACM 33, 3 (March), 338-348.

34. NEIL LAMB, (2008). Copy Number Variation. (Disponible en: http://www.hudsonalpha.org/education/outreach/basics/cnv. Consultado el: 25 de junio del 2013).

35. BIOLOGY COMPUTES, (2012). Pipeline. (Disponible en: http://gtbinf.wordpress.com/2012/12/01/group-5-exome-analysis-project/pipeline-2/. Consultado el: 25 de junio del 2013).

36. ALCALA M, (2007). medida de la usabilidad. (Disponible en: http://www.issi.uned.es/CalidadSoftware/Noticias/PFC_2.pdf. Consultado el: 24 de marzo del 2013).

37. SANGER INSTITUTE, (2013). Sanger method. (Disponible en: http://www.sanger.ac.uk/about/people/biographies/fsanger.html. Consultado el: 2 de marzo del 2013).

38. LIFE SEQUENCING, (2008). La secuenciación 454 es el poder en el que se basa el sistema de secuenciación de Genomas FLX. (Disponible en: http://www.lifesequencing.com/phttp://www.lifesequencing.com/pages/tecnologiaages/tecnologia. Consultado el: 2 de marzo del 2013).

39. BOWTIE 2, (2013). Table of Contents. (Disponible en: http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml. Consultado el: 23 de julio del 2013).

40. DUITAMA J *, QUINTERO J, CRUZ D, QUINTERO C, HUBMANN G, FOULQUIE-MORENO M, VERSTREPEN K J., THEVELEIN J M. AND TOHME J. (2013). An integrated framework for discovery and genotyping of genomic variants from high throughput sequencing experiments.

41. KRUG S. (2006). No me hagas pensar, Una aproximación a la usabilidad en la Web. (Disponible en: http://www.disenomovil.mobi/multimedia_un/01_intro_ux/no_me_hagas_pensar_steve%20_krug_2da%20ed.pdf. Consultado el: 8 de noviembre del 2013).

42. PRESSMAN, ROGER (2002). Ingenieria del Software, un enfoque práctico, Mc-Graw Hill.

43. SOFTWARE ENGINEERING, (2004). ICSE. (Disponible en: http://resources.sei.cmu.edu/asset_files/Presentation/2009_017_001_24441.pdf. Consultado el: 8 de noviembre del 2013).

44. CENTRO DE BIOLOGÍA MOLECULAR, (2011). SECUENCIACIÓN MASIVA Nuevas tecnologías y sus aplicaciones. Madrid España.

45. WEB RADA G. (2007). Estudios de cohortes. (Disponible en: http://escuela.med.puc.cl/Recursos/recepidem/epiAnal3.htm. Consultado el: 15 de Julio del 2013).

46. SNYDER, M., DU, J., & GERSTEIN, M. (2010). Personal genome sequencing: current approaches and challenges. *Genes & Development*, *24*(5), 423–431.

47. WILEY J & SONS. (1996). Practical Software Maintenance. New York.

48. ISO 9126. (1997). The ISO 9126 Standard (Disponible en: http://www.issco.unige.ch/en/research/projects/ewg96/node14.html. Consultado el: 23 de julio de 2013).

ANEXO A

FORMATO DE MATRIZ DE REQUERIMIENTOS FUNCIONALES

Tabla 9: requerimiento número uno.

<table>
<tr><td>ID Req.</td><td>Descripción Requerimiento</td><td>ID Req</td><td>Prioridad</td><td>Fecha

(DD-MMM-YYYY)</td><td>Casos de Uso relacionados</td></tr>
<tr><td>1</td><td>La comparación entre un genoma de referencia y lecturas genómicas esto, con el fin de poder realizar resecuenciación.</td><td>Padre</td><td>Alta</td><td>18-10-2012</td><td>CU_1. Mapear lecturas con respecto a un genoma de referencia.</td></tr>
<tr><td colspan="6">Descripción

El sistema debe permitir comparar un genoma de referencia contra lecturas que provienen de secuenciadores como Illumina y 454. Esto, con el fin de hacer alineamientos de cada lectura para una posición del genoma de referencia.</td></tr>
</table>

Tabla 10: requerimiento número dos.

<table>
<tr><td>ID Req.</td><td>Descripción Requerimiento</td><td>ID Req</td><td>Prioridad</td><td>Fecha

(DD-MMM-YYYY)</td><td>Casos de Uso relacionados</td></tr>
<tr><td>2</td><td>Comparar un archivo BAM, con las lecturas de un genoma y un genoma de referencia, con el fin de encontrar variabilidad genética.</td><td>Padre</td><td>Alta</td><td>01-03-2013</td><td>CU_1. Mapear lecturas con respecto a un genoma de referencia.

CU_2. Ordenar archivo SAM.

CU_4. Encontrar Variantes</td></tr>
<tr><td colspan="6">Descripción

El sistema debe permitir comparar un archivo BAM ordenado contra una referencia con el fin de detectar variaciones diferentes, así como: SNPs, CNVs y Variantes estructurales.</td></tr>
</table>

Tabla 11: requerimiento número tres.

<table>
<tr><td>ID Req.</td><td>Descripción Requerimiento</td><td>ID Req</td><td>Prioridad</td><td>Fecha

(DD-MMM-YYYY)</td><td>Casos de Uso relacionados</td></tr>
<tr><td>3</td><td>Llevar a cabo el ordenamiento de un archivo BAM.</td><td>Padre</td><td>Alta</td><td>18-10-2012</td><td>CU_1. Mapear lecturas con respecto a un genoma de referencia.</td></tr>
<tr><td colspan="6">Descripción

El sistema debe permitir ordenar un archivo BAM para que pueda ser compacto y de fácil acceso para los demás procesos de NGSEP.</td></tr>
</table>

Tabla 12: requerimiento número cuatro.

<table>
<tr><td>ID Req.</td><td>Descripción Requerimiento</td><td>ID Req</td><td>Prioridad</td><td>Fecha

(DD-MMM-YYYY)</td><td>Casos de Uso relacionados</td></tr>
<tr><td>4</td><td>El emparejamiento de pares de lecturas que encajan en una posición de un mismo fragmento secuenciado.</td><td>Padre</td><td>Alta</td><td>18-10-2012</td><td>CU_1. Mapear lecturas con respecto a un genoma de referencia.

CU_2. Ordenar archivo SAM.</td></tr>
<tr><td colspan="6">Descripción

El sistema debe permitir analizar un archivo BAM con lecturas del genoma de un organismo en búsqueda de unir las parejas de lecturas contenidas en él y que coinciden en la misma sección del genoma de acuerdo a una longitud de inserción definida.</td></tr>
</table>

Tabla 13: requerimiento número cinco.

ID Req.	Descripción Requerimiento	ID Req	Prioridad	Fecha (DD-MMM-YYYY)	Casos de Uso relacionados
5	Comparar un catálogo de variantes, un catálogo de anotaciones de genes y un genoma referencia, con el objetivo de buscar posibles variaciones o cambios con respecto al genoma de referencia y como pueden influir en la función de los genes.	Padre	Alta	21-12-2012	CU_1. Mapear lecturas con respecto a un genoma de referencia. CU_2. Ordenar archivo SAM.
Descripción *El sistema debe permitir comparar un catálogo de variantes, un catálogo de anotaciones de genes y un genoma de referencia, con el objetivo de buscar posibles variaciones o cambios con respecto al genoma de referencia y como pueden influir en la función de los genes.*					

Tabla 14: requerimiento número seis.

ID Req.	Descripción Requerimiento	ID Req	Prioridad	Fecha (DD-MMM-YYYY)	Casos de Uso relacionados
6	Llevar a cabo la comparación de lecturas de un genoma (archivo BAM) contra su referencia (archivo Fasta) con el fin de organizar las lecturas alineadas con respecto a la referencia, generando un archivo de salida con los alineamientos únicos y simples de cada lectura.	Padre	Alta	04-01-2013	CU_1. Mapear lecturas con respecto a un genoma de referencia. CU_2. Ordenar archivo SAM.
Descripción *El sistema debe permitir comparar las lecturas presentes en un archivo BAM, con el genoma de referencia del organismo, posteriormente se procede a indicar el número de errores de secuenciación para cada posición del genoma en que se encuentre una lectura. Se debe tener una distribución homogénea alrededor de cada lectura.*					

Tabla 15: requerimiento número siete.

ID Req.	Descripción Requerimiento	ID Req	Prioridad	Fecha (DD-MMM-YYYY)	Casos de Uso relacionados
7	Mezclar tres archivos con variantes y comparar contra la referencia en búsqueda de las posiciones que se encuentran con variación.	Padre	Alta	18-03-2013	CU_1. Mapear lecturas con respecto a un genoma de referencia. CU_2. Ordenar archivo SAM. CU_4. Encontrar Variantes.
Descripción El sistema debe permitir Mezclar tres archivos con variantes y comparar contra la referencia en búsqueda de las posiciones que se encuentran con variación genética y facilitar su análisis desde ancestros al organismo como los padres, generando concordancia de genotipos.					

Tabla 16: requerimiento número ocho.

ID Req.	Descripción Requerimiento	ID Req	Prioridad	Fecha (DD-MMM-YYYY)	Casos de Uso relacionados
8	Determinar la cantidad de lecturas que cubre cada posición del genoma.	Padre	Alta	04-01-2013	CU_1. Mapear lecturas con respecto a un genoma de referencia. CU_2. Ordenar archivo SAM. CU_4. Encontrar Variantes.
Descripción *El sistema debe permitir*, generar un gráfico y un archivo de estadísticas de acuerdo a la muestra ingresad, con el fin de encontrar la cobertura para cada posición del genoma donde hay una lectura alineada, este proceso tiene en cuenta los alineamientos únicos y múltiples.					

CASOS DE USO DEL SISTEMA

CU_1. Mapear lecturas con respecto a un genoma de referencia.

CU_2. Ordenar archivo SAM.

CU_3. Emparejar pares de lecturas.

CU_4. Encontrar Variantes.

CU_5. Identificar el efecto de variaciones en los genes.

CU_6. Mezclar en un solo archivo la información de diferentes muestras analizadas.

CU_7. Cantidad de posiciones cubiertas por el genoma.

CU_8. Qué proporción de llamadas diferentes a la referencia se encuentran.

CU_9. Ingresar archivo FASTQ.

CU_10. Ingresar archivo Fasta.

CU_11. Ingresar archivo BAM organizado.

CU_12.Generar archivo BAM.

CU_13. Generar Log.

CU_14. Generar historial de referencias.

CU_15. Generar archivo VCF.

CU_16. Generar archivo GFF.

CU_17. Generar historial de variants detector.

CU_18. Generar archivo CNV.

CU_19. Generar archivo Coverage.stats

CU_20. Generar historial de GFF.

CU_21. Generar archivo bai.

CU_22. Generar archivo sam pairing.

CU_23. Generar archivo de estadísticas de cobertura.

CU_24. Generar grafica de cobertura.

CU_25. Generar grafica de estadísticas de calidad.

CU_26. Generar archivo vcf con anotaciones de genes.

CU_27. Ingresar archivo de coverage.stats.

CU_28. Ingresar archivo de estadísticas de calidad.

CU_29. Ingresar archivo VCF de variants genómicas.

CU_30. Ingresar archivo gff.

CU_31. Generar VCF con información mezclada de varias muestras y sus correspondientes genotipos.

CU_32. Ingresar archivo de historial variants detector.

CU_33. Generar archivo SAM.

CU_34. Generar VCF con información mezclada de varias muestras.

DIAGRAMA DE CASOS DE USO

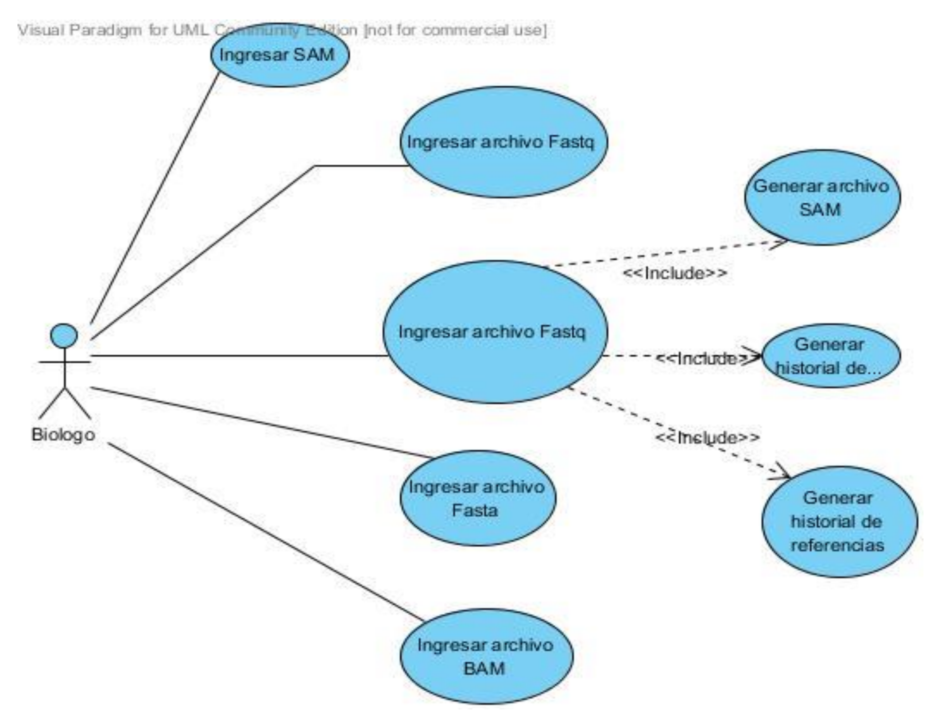


Ilustración 98: Diagrama de caso de uso de NGSEP; generar archivo Sam, ingresar archivo Fastq, generar historial de referencias.

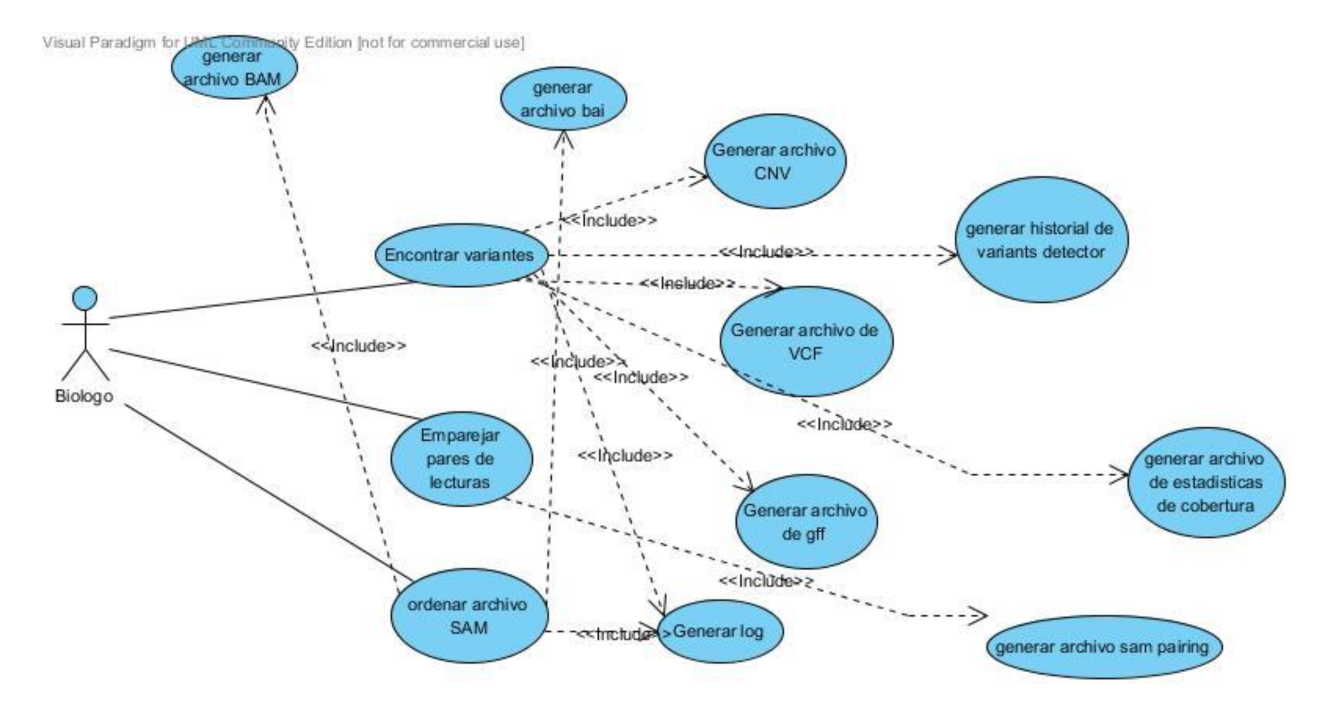


Ilustración 99: Diagrama de casos de uso de NGSEP; encontrar variantes, ordenar archivo SAM, generar archivo VCG, GFFF, CNV.

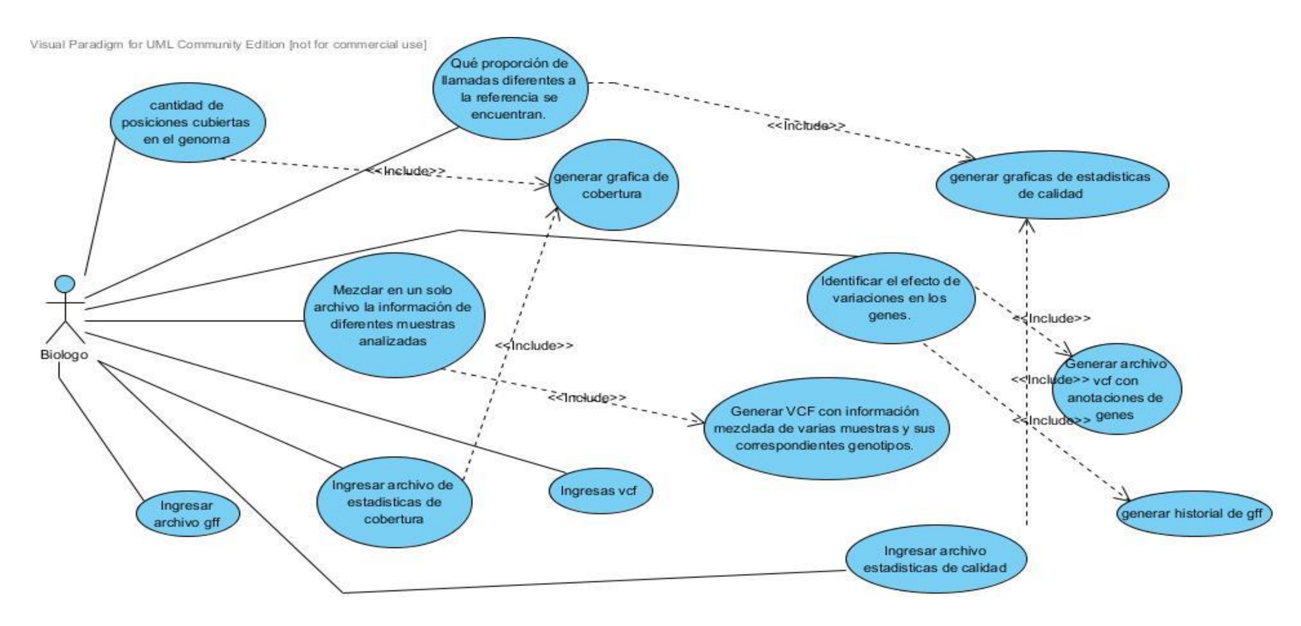


Ilustración 100: Diagrama de casos de uso de NGSEP; generar graficas de cobertura, generar historial de GFF, mezclar en un solo archivo información de diferentes muestras analizadas.

DIAGRAMA DE CLASES

En este apartado, se muestra una parte del diagrama total de clases de NGSEP Ilustración 101 ya que se presenta mucha dificultad a la hora de mostrar el diagrama en su totalidad porque, actualmente NGSEP posee más de 40 clases.

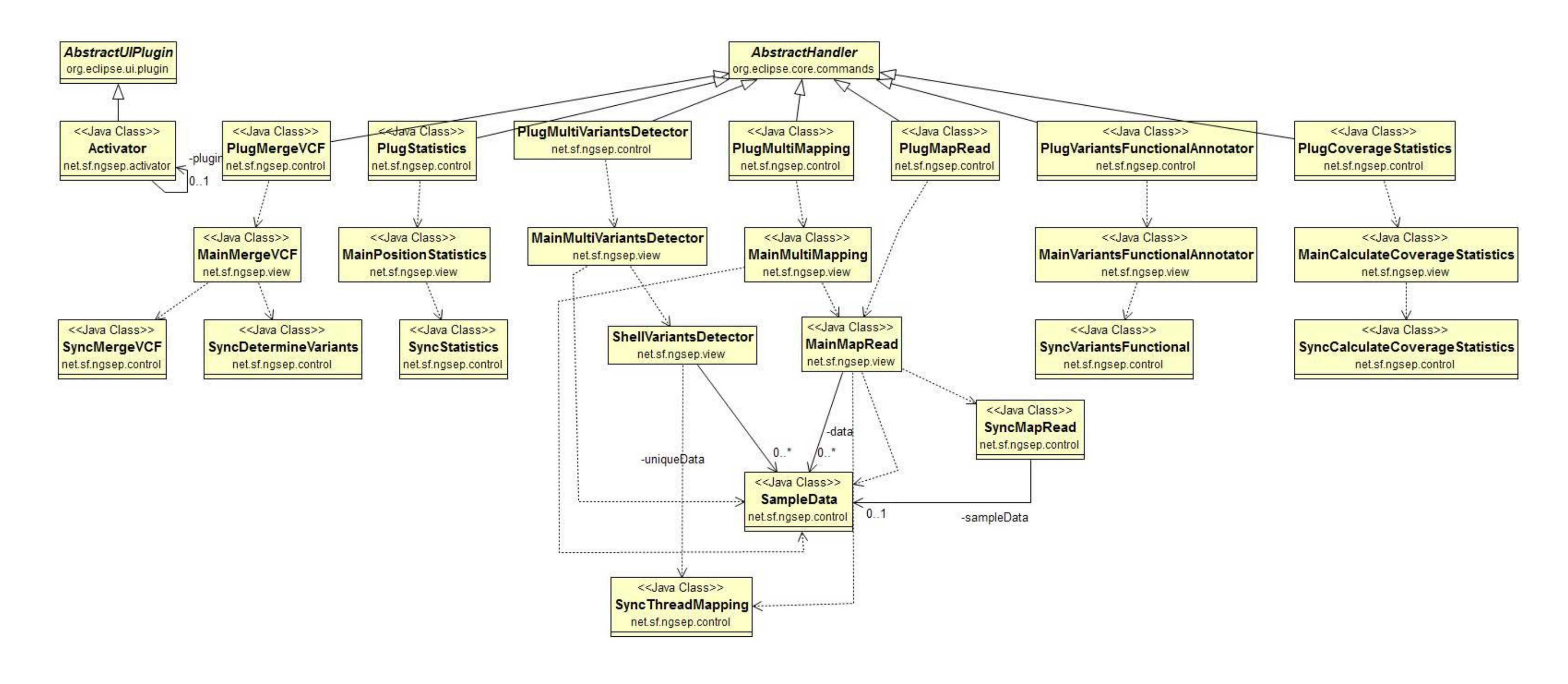


Ilustración 101: Diagrama de clases de NGSEP[1].

[1] Este diagrama es un extracto del diagrama original de NGSEP.

GUIONES

A continuación, se presentan los guiones para los casos de uso más relevantes del Plug-in NGSEP.

GUION CASO DE USO 1

<table>
<tr><td>No.</td><td>CU_1. Mapear lecturas con respecto a un genoma de referencia.</td></tr>
<tr><td>Nombre</td><td>Mapear lecturas con respecto a un genoma de referencia.</td></tr>
<tr><td>Descripción</td><td>La función de este caso de uso es alinear cada una de las lecturas ingresadas por el usuario en una posición del genoma de referencia.</td></tr>
<tr><td>Actores</td><td>Biólogo.</td></tr>
<tr><td>Fase</td><td>Análisis</td></tr>
<tr><td>Guion</td><td>
<table>
<tr><th>Actor</th><th>Sistema</th></tr>
<tr><td>1. Crea un general Project en Eclipse.</td><td></td></tr>
<tr><td>2. Copia las lecturas de secuencias en las que desea encontrar variantes genómicas con respecto al genoma de referencia y las pega en el proyecto creado, estas lecturas deben estar en formato FASTAQ.</td><td></td></tr>
<tr><td>3. Copia el genoma de referencia con el que se van a comparar las lecturas, este archivo debe estar en formato Fasta.</td><td></td></tr>
<tr><td>4. Selecciona una o dos lecturas, si selecciona dos lecturas estas deben ser complemento una de la otra.</td><td></td></tr>
<tr><td>5. Luego de seleccionar las lecturas o lectura, da clic derecho sobre cualquiera de las selecciones y busca la</td><td></td></tr>
</table>
</td></tr>
</table>

	opción NGSEP Menu dentro de la ventana desplegada al lado derecho de la selección.		
	6. Una vez encontrado el menú de NGSEP, ubica el puntero encima del menú.	7. El sistema valida la ubicación del puntero y procede a mostrar una serie de submenús.	
	8. El usuario ubica la opción de NGSEP llamada Map Reads y da clic sobre esta.	9. El sistema valida el clic y despliega la pantalla de Map Reads, con las rutas de las lecturas cargadas en las cajas de texto de nombre file #1 y file # 2 en caso de que el usuario seleccione 2 lecturas si no solo carga file#1.	
	10. El usuario ubica la entrada para el genoma de referencia, y da clic en el botón de cargar.	11. El sistema valida el clic, y despliega un wizard para cargar la ruta donde se encuentra el genoma de referencia.	
	12. El usuario elige el genoma de referencia y carga la ruta en el wizard desplegado por NGSEP.	13. El sistema valida la ruta del genoma de referencia y pinta la ruta en la caja de texto que acompaña el titulo reference.	
	14. El usuario carga la ruta donde desea generar el archivo de salida.		
	15. El usuario ingresa las opciones que deseé de más y que estén comprendidas en la pantalla de Map Reads.	16. El sistema valida los valores en las entradas de la pantalla Map Reads y comienza la ejecución.	
		17. El sistema activa una barra de progreso en la vista de NGSEP, esta barra marcara el avance del proceso, indicando cuando comienza y cuando finaliza, de igual forma permite terminar el proceso si el usuario lo desea dándole clic en el botón rojo que acompaña la barra.	
		18. El sistema alinear cada una de las lecturas en una posición del genoma de referencia.	
		19. El sistema genera un log en la raíz del proyecto con información relevante a la ejecución actual del proceso Map Reads que envió el usuario.	
		20. El sistema crea un archivo de historial para la referencia usada como genoma de referencia.	
		21. El sistema genera un archivo SAM	

		en la ruta donde se encuentran las lecturas, con el resultado de los alineamientos.
		22. El sistema da por finalizado el proceso y desaparece la barra de progreso.
Excepciones	1. Si no se ingresa referencia.	

Nombre	Mensaje
Paso 1	
	1. Si no se carga una ruta en la caja de texto de referencia el sistema despliega el mensaje “campo referencia obligatorio”.
	2. Vuelve al paso 1.

2. Si se borra las rutas de las lecturas.

Nombre	Mensaje
Paso 2	
	1. Si se borra la ruta de las lecturas en las cajas de texto de file #1 y file #2, el sistema despliega el mensaje “campo obligatorio file#1 o file#2”.
	2. Vuelve al paso 2.

3. Si no ingresa archivo de salida.

Nombre	Mensaje
Paso 3	
	1. Si se borra la ruta del archivo de salida o no se ingresa en las cajas de texto de output file, el sistema despliega el mensaje “campo obligatorio Output File”.
	2. Vuelve al paso 3.

4. Si el formato de las lecturas esta errado.

Nombre	Mensaje
Paso 4	
	1. Si el formato de las lecturas esta errado se genera un mensaje de error en el archivo log creado por la aplicación, de igual forma la barra de progreso no arranca su ejecución.
	2. Vuelve al paso 4.

<table>
<tr><td></td><td>5. Si el formato del genoma de referencia esta errado.
<table>
<tr><th>Nombre</th><th>Mensaje</th></tr>
<tr><td>Paso 5</td><td></td></tr>
<tr><td></td><td>1. Si el formato del genoma de referencia esta errado se genera un mensaje de error en el archivo log creado por la aplicación, de igual forma la barra de progreso no arranca su ejecución.</td></tr>
<tr><td></td><td>2. Vuelve al paso 5.</td></tr>
</table></td></tr>
<tr><td>Casos de uso relacionados</td><td>CU_9. Ingresar archive FASTQ.
CU_13. Generar Log.
CU_33. Generar archivo SAM.
CU_10. Ingresar archivo Fasta.
CU_14. Generar historial de referencias.</td></tr>
<tr><td>Requerimiento Fuente</td><td>El sistema debe de permitir: La comparación entre un genoma de referencia y lecturas genómicas esto, con el fin de poder realizar resecuenciación.</td></tr>
<tr><td>Autor</td><td>Juan Camilo Quintero</td></tr>
<tr><td>Fecha Creación</td><td>Octubre 18 del 2012</td></tr>
<tr><td>Fecha de Ultima Modificación</td><td>Octubre 18 del 2012</td></tr>
</table>

GUION CASO DE USO 2

<table>
<tr><td>No.</td><td>CU_2. Ordenar archivo SAM</td></tr>
<tr><td>Nombre</td><td>Ordenar archivo SAM.</td></tr>
<tr><td>Descripción</td><td>La función de este caso de uso es ordenar el archivo SAM producto de la ejecución del caso de uso 1.</td></tr>
<tr><td>Actores</td><td>Biólogo.</td></tr>
<tr><td>Fase</td><td>Análisis</td></tr>
<tr><td>Guion</td><td>
<table>
<tr><th>Actor</th><th>Sistema</th></tr>
<tr><td>.</td><td></td></tr>
<tr><td>1. Selecciona un archivo SAM con alineamientos de lecturas con respecto a una posición del genoma de referencia.</td><td></td></tr>
<tr><td>2. Luego de seleccionar el archivo SAM, da clic derecho sobre este y busca la opción NGSEP Menu dentro de la ventana desplegada al lado derecho de la selección.</td><td></td></tr>
<tr><td>3. Una vez encontrado el menú de NGSEP, ubica el puntero encima del menú.</td><td></td></tr>
<tr><td></td><td>4. El sistema valida la ubicación del puntero y procede a mostrar una serie de submenús.</td></tr>
<tr><td>5. El usuario ubica la opción de NGSEP llamada Sort Alignmet y da clic sobre esta.</td><td>6. El sistema valida el clic y despliega la pantalla de Sort Alignmet, con la ruta del archivo SAM seleccionado cargada en la caja de texto que acompaña a la entrada “(*) File (SAM).</td></tr>
<tr><td></td><td>7. El sistema sugiere un archivo de salida con la misma ruta y nombre del archivo de entrada pero con la agregación sorted.bam.</td></tr>
<tr><td>8. El usuario da clic en el botón Sort Alignmet</td><td>9. valida el clic y da comienzo a la ejecución.</td></tr>
<tr><td></td><td>10. Crea archivo log.</td></tr>
<tr><td></td><td>11. Ordena archivo SAM para comprimirlo en un archivo de menor tamaño y que sea de entendimiento para la máquina.</td></tr>
</table>
</td></tr>
</table>

	12. Genera un archivo BAM a partir del SAM ordenado.
Excepciones	1. Si no se ingresa archivo SAM. **Nombre** / **Mensaje** Paso 1 1. Si no se carga una ruta en la caja de texto de File (Sam) el sistema despliega el siguiente mensaje “campo File (Sam) obligatorio”. 2. Vuelve al paso 1. 2. Si se borra o no se ingresa la ruta del archivo de salida. **Nombre** / **Mensaje** Paso 2 1. Si se borra la ruta del archivo de salida, el sistema despliega el mensaje “campo output file obligatorio”. 2. Vuelve al paso 2. 3. Si el archivo SAM no está el formato adecuado. **Nombre** / **Mensaje** Paso 3 1. Si el archivo SAM no está en formato adecuado el archivo log genera una excepción. 2. Vuelve al paso 3.
Casos de uso relacionados	CU_12.Generar archivo BAM. CU_13. Generar Log.
Requerimiento Fuente	El sistema debe permitir: Llevar a cabo el ordenamiento de un archivo BAM..
Autor	Juan Camilo Quintero
Fecha Creación	Octubre 18 del 2012
Fecha de Ultima Modificación	Octubre 18 del 2012

GUION CASO DE USO 4

No.	CU_4. Encontrar Variantes.
Nombre	Encontrar Variantes.
Descripción	La función de este caso de uso es, comparar un archivo BAM con lecturas secuenciadas que se encuentran alineadas contra el genoma de referencia del organismo secuenciado en las lecturas, esta comparación se realiza con el fin de encontrar variantes genómicas presentes en la secuencia de las lecturas.
Actores	Biólogo.
Fase	Análisis
Guión	(ver tabla)

Actor	Sistema
1. Selecciona un archivo BAM.	
2. Luego de seleccionar el archivo BAM, da clic derecho sobre este y busca la opción NGSEP Menu dentro de la ventana desplegada al lado derecho de la selección.	
3. Una vez encontrado el menú de NGSEP, ubica el puntero encima del menú.	
	4. El sistema valida la ubicación del puntero y procede a mostrar una serie de submenús.
5. El usuario ubica la opción de NGSEP llamada Find Variants.	6. El sistema valida el clic y despliega la pantalla de Find Variants, con la ruta del archivo BAM seleccionado cargada en la caja de texto que acompaña a la entrada “(*) File.
	7. El sistema sugiere un archivo de salida con la misma ruta y nombre del archivo de entrada pero con la extensión vcf.
8. Ingresa la ruta donde se encuentra el archivo del genoma de referencia.	

	9. Ingresa demás opciones si así lo desea y si están comprendidas en la pantalla de Find Variants.	
	10. Da clic en el botón Find Variants.	11. El sistema valida las entradas y comienza la ejecución.
		12. Crea una barra de progreso en la vista de procesos de NGSEP, esta barra de progreso indica el avance de la ejecución del proceso actual lanzado en Find Variants, de igual forma permite finalizar el proceso si el usuario lo desea dando clic en el botón rojo que acompaña la barra de progreso.
		13. Compara el genoma de referencia con el archivo BAM posición a posición con el fin de encontrar diferencias en las secuencias, una vez encontradas estas diferencias la clasifica de acuerdo al tipo de variación genómica que es y las ingresa un archivo que se genera como salida con extensión vcf el cual contiene las variaciones en el cromosoma que ocurrió, la posición inicial y final del genoma donde encontró la variación, el nombre de la variación, el cambio que ocurrió en los nucleótidos y el genotipo de la muestra donde está la variante.
		14. Genera un archivo log con el registro de la ejecución del proceso actual de Find Variants.
		15. Genera un archivo con el historial de la referencia usada como genoma de referencia.
		16. Genera un archivo con los CNVs detectados en las posiciones del genoma.
		17. Genera un archivo GFF.
		18. Genera un archivo con el historial del genoma de referencia usado, el vcf generado y la muestra ingresada (archivo BAM).
		19. Finaliza la ejecución y elimina la barra de progreso.
Excepciones	1. Si no se ingresa archivo BAM.	

Nombre	Mensaje
Paso 1	

	1. Si no se carga una ruta en la caja de texto de File el sistema despliega el siguiente mensaje “campo File obligatorio”.
	2. Vuelve al paso 1.

2. Si se borra o no se ingresa la ruta del archivo de salida.

Nombre	Mensaje
Paso 2	
	1. Si se borra la ruta del archivo de salida, el sistema despliega el mensaje “campo output file obligatorio”.
	2. Vuelve al paso 2.

3. Si el archivo BAM no está el formato adecuado.

Nombre	Mensaje
Paso 3	
	1. Si el archivo SAM no está en formato adecuado el archivo log genera una excepción.
	2. Vuelve al paso 3.

4. Si no se ingresa el genoma de referencia.

Nombre	Mensaje
Paso 4	
	1. Si no es ingresa el genoma de referencia el sistema despliega el mensaje de excepción “campo reference file obligatorio”.
	2. Vuelve al paso 4.

Casos de uso relacionados	CU_13. Generar Log. CU_15. Generar archivo VCF. CU_16. Generar archivo GFF. CU_17. Generar historial de variants detector. CU_18. Generar archivo CNV. CU_10. Ingresar archivo Fasta.

	CU_11. Ingresar archivo BAM organizado. CU_14. Generar historial de referencias.
Requerimiento Fuente	El sistema debe permitir: Comparar un archivo BAM, con las lecturas de un genoma y un genoma de referencia, con el fin de encontrar variabilidad genética.
Autor	Juan Camilo Quintero
Fecha Creación	Enero 03 del 2013
Fecha de Ultima Modificación	Enero 03 del 2013

GUION CASO DE USO 5

No.	CU_5. Identificar el efecto de variaciones en los genes.
Nombre	Identificar el efecto de variaciones en los genes.
Descripción	La función de este caso de uso es, comparar un archivo vcf con variantes genómicas, un catálogo de genes y un genoma de referencia pertenecientes a la secuencia de un organismo con el fin de generar un archivo que contenga si las variantes encontradas en el caso de uso cuatro tiene un efecto en los genes de dicho organismo.
Actores	Biólogo.
Fase	Análisis
Guión	<table><tr><th>Actor</th><th>Sistema</th></tr><tr><td>1. Selecciona un archivo VCF con variantes genómicas.</td><td></td></tr><tr><td>2. Luego de seleccionar el archivo VCF, da clic derecho sobre este y busca la opción NGSEP Menu dentro de la ventana</td><td></td></tr></table>

	desplegada al lado derecho de la selección.		
	3. Una vez encontrado el menú de NGSEP, ubica el puntero encima del menú.		
		4. El sistema valida la ubicación del puntero y procede a mostrar una serie de submenús.	
	5. El usuario ubica la opción de NGSEP llamada Variants Functional Annotation.	6. El sistema valida el clic y despliega la pantalla de Variants Functional Annotation, con la ruta del archivo VCF seleccionado cargada en la caja de texto que acompaña a la entrada "(VCF) Variants File.	
		7. El sistema sugiere un archivo de salida con la misma ruta y nombre del archivo de entrada pero la agregación Annotated.	
	8. Ingresa la ruta donde se encuentra el archivo del genoma de referencia.		
	9. Ingresa el catálogo de genes (archivo GFF) de la especie del genoma de referencia.		
	10. Da clic en el botón Variants Functional.	11. El sistema valida las entradas y comienza la ejecución.	
		12. Crea una barra de progreso en la vista de procesos de NGSEP, esta barra de progreso indica el avance de la ejecución del proceso actual lanzado en Variants Functional Annotation, de igual forma permite finalizar el proceso si el usuario lo desea dando clic en el botón rojo que acompaña la barra de progreso.	
		13. Compara el genoma de referencia con el archivo VCF y el catálogo de genes con el fin de detectar el efecto en los genes de las variantes.	
		14. Genera un archivo log con el registro de la ejecución del proceso actual de Variants Functional Annotation.	
		15. Genera un archivo con el historial de la referencia usada como genoma de referencia.	
		16. Genera un archivo de historial con el GFF usado como catálogo de genes.	

		17. Genera un VCF con las variantes, la posición en el genoma y la región o calificativo del gen que afecta.
		18. Finaliza la ejecución y elimina la barra de progreso.

Excepciones

1. Si no se ingresa archivo VCF.

Nombre	Mensaje
Paso 1	
	1. Si no se carga una ruta en la caja de texto de File el sistema despliega el siguiente mensaje "campo (VCF) Variants File obligatorio".
	2. Vuelve al paso 1.

2. Si se borra o no se ingresa la ruta del archivo de salida.

Nombre	Mensaje
Paso 2	
	5. Si se borra la ruta del archivo de salida, el sistema despliega el mensaje "campo output file obligatorio".
	6. Vuelve al paso 2.

3. Si no se ingresa el genoma de referencia.

Nombre	Mensaje
Paso 3	
	1. Si no es ingresa el genoma de referencia el sistema despliega el mensaje de excepción "campo reference file obligatorio".
	2. Vuelve al paso 3

4. Si no se ingresa archivo GFF.

Nombre	Mensaje
Paso 4	
	1. Si no se carga una ruta en la caja de texto de File el sistema despliega el siguiente mensaje "campo (GFF) Gene Annotation File".
	2. Vuelve al paso 4.

Casos de uso relacionados	CU_13. Generar Log. CU_10. Ingresar archivo Fasta. CU_14. Generar historial de referencias. CU_20. Generar historial de GFF. CU_26. Generar archivo vcf con anotaciones de genes. CU_29. Ingresar archivo VCF de variants genómicas. CU_30. Ingresar archivo GFF.
Requerimiento Fuente	El sistema debe permitir: Comparar un catálogo de variantes, un catálogo de anotaciones de genes y un genoma referencia, con el objetivo de buscar posibles variaciones o cambios con respecto al genoma de referencia y como pueden influir en la función de los genes.
Autor	Juan Camilo Quintero
Fecha Creación	Diciembre 21 del 2012
Fecha de Ultima Modificación	Diciembre 21 del 2012

GUION CASO DE USO 6

<table>
<tr><td>No.</td><td>CU_6. Mezclar en un solo archivo la información de diferentes muestras analizadas.</td></tr>
<tr><td>Nombre</td><td>Mezclar en un solo archivo la información de diferentes muestras analizadas.</td></tr>
<tr><td>Descripción</td><td>La función de este caso de uso es, este proceso se divide en dos fases: la primera tiene por objeto determinar la lista de las variantes encontradas en al menos uno de los archivos VCF que se generaron en el proceso de detección de variantes, posteriormente generar un archivo VCF común entre las muestras seleccionadas. Después, el proceso requiere la ejecución de nuevo de variants Detector para todas las muestras seleccionadas, pero utilizando el archivo común generado anteriormente, este proceso genera nuevos archivos VCF. Por último, podrá fusionar esos nuevos archivos VCF que se generaron en uno solo, que muestra la herencia de padres a hijos en los alelos.</td></tr>
<tr><td>Actores</td><td>Biólogo.</td></tr>
<tr><td>Fase</td><td>Análisis</td></tr>
<tr><td>Guión</td><td>
<table>
<tr><th>Actor</th><th>Sistema</th></tr>
<tr><td>1. Selecciona el historial de variants detector.</td><td></td></tr>
<tr><td>2. Luego de seleccionar el historial de variants detector, da clic derecho sobre este y busca la opción NGSEP Menu dentro de la ventana desplegada al lado derecho de la selección.</td><td></td></tr>
<tr><td>3. Una vez encontrado el menú de NGSEP, ubica el puntero encima del menú.</td><td></td></tr>
<tr><td></td><td>4. El sistema valida la ubicación del puntero y procede a mostrar una serie de submenús.</td></tr>
<tr><td>5. El usuario ubica la opción de NGSEP llamada Merge VCF.</td><td>6. El sistema valida el clic y despliega la pantalla de Merge VCF, con una tabla de cinco columnas y con los registros del número de muestras ejecutadas en variants detector, organizando la tabla con el nombre de la muestra (archivo BAM), genoma de referencia y archivo generado con variantes (VCF), adicionalmente se crear un combo box que va permitir al usuario seleccionar si quiere mezclar la muestra.</td></tr>
</table>
</td></tr>
</table>

	7. El sistema sugiere un archivo de salida con la misma ruta y nombre del archivo de entrada pero la agregación de MergeFile.vcf.
8. Selecciona las muestras que desea mezclar.	
9. Da clic en el botón determine list of variants.	10. Valida el número de muestras seleccionadas.
	11. Mezcla todas las variantes comunes en las muestras seleccionadas y crea un archivo VCF con todas estas variantes.
	12. Genera barra de progreso que indica el avance del proceso y elimina la barra cuando finaliza.
13. Una vez finalizado el proceso de determinar la lista de variantes, selecciona las muestras que selecciono en la pantalla de Merge e ingresa a variants detector y lo ejecuta de nuevo pero ingresado el archivo común en la entrada know variants File.	14. Valida entradas en variants detector y procede a detectar las variantes comunes con su respectivo genotipo.
	15. Genera archivo VCF con variantes comunes y sus genotipos.
	16. Genera archivo log.
	17. Genera archivo CNV.
	18. Crea barra de progreso que indica avance de principio a fin.
19. Una vez termine de correr variants detector por cada una de las muestras que selecciono para mezclar, selecciona el historial generado por variants detector y le da clic derecho y procede a ingresar de nuevo a Merge VCF.	20. Crea de nuevo la tabla con las 4 columnas y la información del historial de variants detector.
21. Selecciona las muestras que desea mezclar.	22. Valida las muestras seleccionadas y crea un archivo VCF con todas las muestras seleccionadas mezcladas con sus respectivos genotipos.
	23. Genera barra de progreso de principio a fin de la ejecución del proceso.
	24. Genera archivo log.

<table>
<tr>
<td>Excepciones</td>
<td>
1. Si no se ingresa archivo BAM.
<table>
<tr><th>Nombre</th><th>Mensaje</th></tr>
<tr><td>Paso 1</td><td></td></tr>
<tr><td></td><td>3. Si no se carga una ruta en la caja de texto de File el sistema despliega el siguiente mensaje “campo File obligatorio”.</td></tr>
<tr><td></td><td>4. Vuelve al paso 1.</td></tr>
</table>
2. Si se borra o no se ingresa la ruta del archivo de salida.
<table>
<tr><th>Nombre</th><th>Mensaje</th></tr>
<tr><td>Paso 2</td><td></td></tr>
<tr><td></td><td>7. Si se borra la ruta del archivo de salida, el sistema despliega el mensaje “campo output file obligatorio”.</td></tr>
<tr><td></td><td>8. Vuelve al paso 2.</td></tr>
</table>
3. Si no se ingresa el genoma de referencia.
<table>
<tr><th>Nombre</th><th>Mensaje</th></tr>
<tr><td>Paso 3</td><td></td></tr>
<tr><td></td><td>3. Si no es ingresa el genoma de referencia el sistema despliega el mensaje de excepción “campo reference file obligatorio”.</td></tr>
<tr><td></td><td>4. Vuelve al paso 3</td></tr>
</table>
4. Si no selecciona ninguna muestra.
<table>
<tr><th>Nombre</th><th>Mensaje</th></tr>
<tr><td>Paso 4</td><td></td></tr>
<tr><td></td><td>1. Si no se selecciona ninguna muestra en la pantalla Merge VCF, se despliega el siguiente mensaje “Al menos debe seleccionar dos muestras de manera obligatoria para ejecutar este proceso”.</td></tr>
<tr><td></td><td>2. Vuelve al paso 4.</td></tr>
</table>
</td>
</tr>
</table>

Casos de uso relacionados	CU_13. Generar Log. CU_10. Ingresar archivo Fasta. CU_14. Generar historial de referencias. CU_4. Encontrar Variantes. CU_15. Generar archivo VCF. CU_16. Generar archivo GFF. CU_17. Generar historial de variants detector. CU_18. Generar archivo CNV. CU_11. Ingresar archivo BAM organizado. CU_32. Ingresar archivo de historial variants detector. CU_34. Generar VCF con información mezclada de varias muestras. CU_31. Generar VCF con información mezclada de varias muestras y sus correspondientes genotipos.
Requerimiento Fuente	El sistema debe permitir: Mezclar tres archivos con variantes y comparar contra la referencia en búsqueda de las posiciones que se encuentran con variación.
Autor	Juan Camilo Quintero
Fecha Creación	Marzo 18 del 2013
Fecha de Ultima Modificación	Marzo 18 del 2013

GUION CASO DE USO 7

<table>
<tr><td>No.</td><td>CU_7. Cantidad de posiciones cubiertas por el genoma.</td></tr>
<tr><td>Nombre</td><td>Cantidad de posiciones cubiertas por el genoma.</td></tr>
<tr><td>Descripción</td><td>La función de este caso de uso es, generar un gráfico y un archivo de estadísticas de acuerdo a la muestra ingresad, con el fin de encontrar la cobertura para cada posición del genoma donde hay una lectura alineada, este proceso tiene en cuenta los alineamientos únicos y múltiples.</td></tr>
<tr><td>Actores</td><td>Biólogo.</td></tr>
<tr><td>Fase</td><td>Análisis</td></tr>
<tr><td>Guión</td><td>
<table>
<tr><th>Actor</th><th>Sistema</th></tr>
<tr><td>1. Selecciona el archivo BAM.</td><td></td></tr>
<tr><td>2. Luego de seleccionar el archivo BAM, da clic derecho sobre este y busca la opción NGSEP Menu dentro de la ventana desplegada al lado derecho de la selección.</td><td></td></tr>
<tr><td>3. Una vez encontrado el menú de NGSEP, ubica el puntero encima del menú.</td><td></td></tr>
<tr><td></td><td>4. El sistema valida la ubicación del puntero y procede a mostrar una serie de submenús.</td></tr>
<tr><td>5. El usuario ubica la opción de NGSEP llamada Calculated Coverage Statistics.</td><td>6. El sistema valida el clic y despliega la pantalla de Calculated Coverage Statistics, con la ruta del archivo seleccionado y con una sugerencia para un archivo de salida.</td></tr>
<tr><td>7. Selecciona si desea el grafico con múltiples alineamientos, si no selecciona esta opción el sistema toma por defecto alineamientos únicos.</td><td></td></tr>
<tr><td>8. Da clic en el botón statistics.</td><td>9. Valida las entradas.</td></tr>
<tr><td></td><td>10. Buscan en el archivo BAM las lecturas que tengan alineaciones con cobertura superior o igual a 50 pb (pares bases).</td></tr>
<tr><td></td><td>11. Genera barra de progreso que indica el avance del proceso y elimina la barra cuando finaliza.</td></tr>
<tr><td></td><td>12. Genera archivo log.</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td></td><td><table><tr><td></td><td>13. Genera grafica de cobertura con alineamientos únicos o múltiples de acuerdo a lo seleccionado por el usuario.</td></tr><tr><td></td><td>14. Genera archivo de estadísticas con la cobertura y el número de alineamientos en cada posición del genoma donde esta una lectura.</td></tr></table></td></tr>
<tr><td>Excepciones</td><td>1. Si no se ingresa archivo BAM.
<table><tr><th>Nombre</th><th>Mensaje</th></tr><tr><td>Paso 1</td><td></td></tr><tr><td></td><td>1. Si no se carga una ruta en la caja de texto de File el sistema despliega el siguiente mensaje “campo File obligatorio”.</td></tr><tr><td></td><td>2. Vuelve al paso 1.</td></tr></table>
2. Si se borra o no se ingresa la ruta del archivo de salida.
<table><tr><th>Nombre</th><th>Mensaje</th></tr><tr><td>Paso 2</td><td></td></tr><tr><td></td><td>1. Si se borra la ruta del archivo de salida, el sistema despliega el mensaje “campo output file obligatorio”.</td></tr><tr><td></td><td>2. Vuelve al paso 2.</td></tr></table></td></tr>
<tr><td>Casos de uso relacionados</td><td>CU_13. Generar Log.
CU_11. Ingresar archivo BAM organizado.
CU_19. Generar archivo Coverage.stats
CU_24. Generar grafica de cobertura.</td></tr>
<tr><td>Requerimiento Fuente</td><td>El sistema debe permitir: Determinar la cantidad de lecturas que cubre cada posición del genoma.</td></tr>
<tr><td>Autor</td><td>Juan Camilo Quintero</td></tr>
<tr><td>Fecha Creación</td><td>Enero 4 del 2013</td></tr>
<tr><td>Fecha de Ultima Modificación</td><td>Enero 4 del 2013</td></tr>
</table>

GUION CASO DE USO 8

No.	CU_8. Qué proporción de llamadas diferentes a la referencia se encuentran.
Nombre	Qué proporción de llamadas diferentes a la referencia se encuentran.
Descripción	La función de este caso de uso es, analizar un archivo BAM con lecturas del genoma de un organismo en búsqueda de unir las parejas de lecturas contenidas en él y que coinciden en la misma sección del genoma de acuerdo a una longitud de inserción definida por usuario.
Actores	Biólogo.
Fase	Análisis
Guión	(ver tabla siguiente)

Actor	**Sistema**
1. Selecciona el archivo BAM.	
2. Luego de seleccionar el archivo BAM, da clic derecho sobre este y busca la opción NGSEP Menu dentro de la ventana desplegada al lado derecho de la selección.	
3. Una vez encontrado el menú de NGSEP, ubica el puntero encima del menú.	
	4. El sistema valida la ubicación del puntero y procede a mostrar una serie de submenús.
5. El usuario ubica la opción de NGSEP llamada Calculated Quality Statistics.	6. El sistema valida el clic y despliega la pantalla de Calculated Quality Statistics, con la ruta del archivo seleccionado y con una sugerencia para un archivo de salida.
7. Selecciona si desea el grafico con múltiples alineamientos, si no selecciona esta opción el sistema toma por defecto alineamientos únicos.	
8. Ingresa el tamaño de las lecturas si lo conoce, de no ser así el sistema captura por defecto 50	

	9. Da clic en el botón statistics.	10. Valida las entradas.	
		11. Buscan en el archivo BAM las parejas de lecturas que coincidan en la misma posición del genoma y que tenga igual tamaño de inserción.	
		12. Genera barra de progreso que indica el avance del proceso y elimina la barra cuando finaliza.	
		13. Genera archivo log.	
		14. Genera grafica de calidad con alineamientos únicos o múltiples de acuerdo a lo seleccionado por el usuario.	
		15. Genera archivo de estadísticas con la calidad y el número de alineamientos en cada posición del genoma donde esta una lectura.	

Excepciones

1. Si no se ingresa archivo BAM.

Nombre	Mensaje
Paso 1	
	1. Si no se carga una ruta en la caja de texto de File el sistema despliega el siguiente mensaje “campo File obligatorio”.
	2. Vuelve al paso 1.

2. Si se borra o no se ingresa la ruta del archivo de salida.

Nombre	Mensaje
Paso 2	
	1. Si se borra la ruta del archivo de salida, el sistema despliega el mensaje “campo output file obligatorio”.
	2. Vuelve al paso 2.

3. Si se ingresa un tipo de dato diferente a un entero el campo read lenght.

Nombre	Mensaje
Paso 3	
	1. Si se ingresa un dato diferente a entero en el campo read lenght, el sistema despliega el mensaje “campo read lenght solo recibe tipo de datos enteros”.

		2. Vuelve al paso 3.
Casos de uso relacionados	CU_13. Generar Log. CU_11. Ingresar archivo BAM organizado. CU_23. Generar archivo de estadísticas de cobertura. CU_25. Generar grafica de estadísticas de calidad.	
Requerimiento Fuente	El sistema debe permitir: Determinar la cantidad de lecturas que cubre cada posición del genoma.	
Autor	Juan Camilo Quintero	
Fecha Creación	Octubre 18 del 2012	
Fecha de Ultima Modificación	Octubre 18 del 2012	

DIAGRAMAS DE SECUENCIA

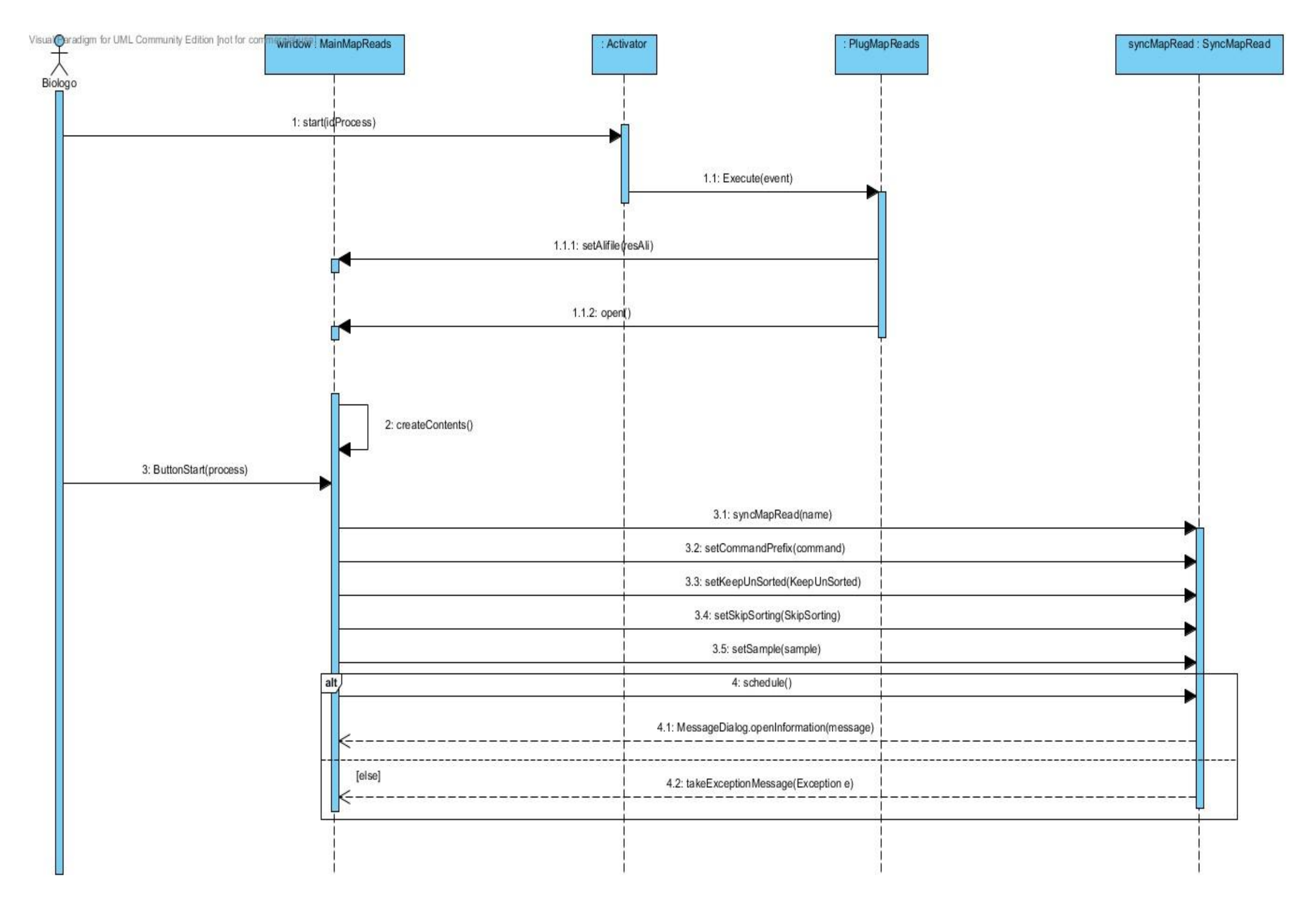
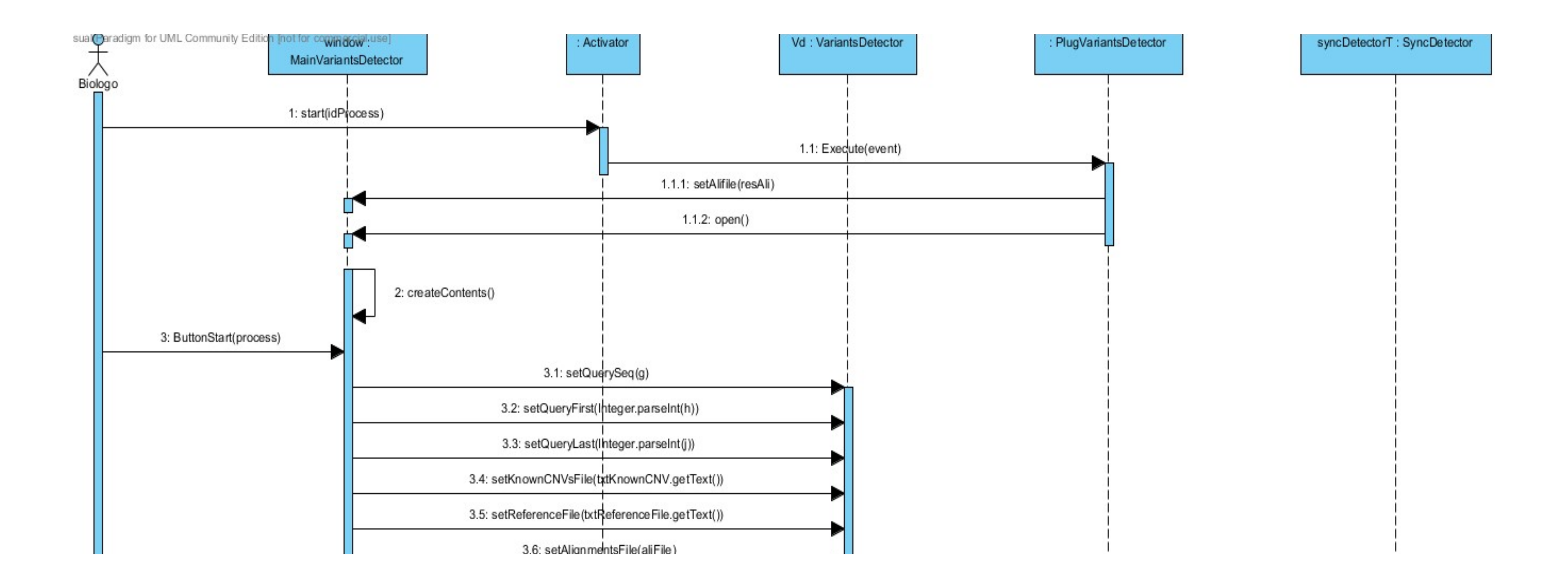


Ilustración 102: Diagrama de secuencia Mapear lecturas con respecto a un genoma de referencia.


Biologo
window : MainVariantsDetector
: Activator
Vd : VariantsDetector
: PlugVariantsDetector
syncDetectorT : SyncDetector
1: start(idProcess)
1.1: Execute(event)
1.1.1: setAlifile(resAli)
1.1.2: open()
2: createContents()
3: ButtonStart(process)
3.1: setQuerySeq(g)
3.2: setQueryFirst(Integer.parseInt(h))
3.3: setQueryLast(Integer.parseInt(j))
3.4: setKnownCNVsFile(txtKnownCNV.getText())
3.5: setReferenceFile(txtReferenceFile.getText())
3.6: setAlignmentsFile(aliFile)

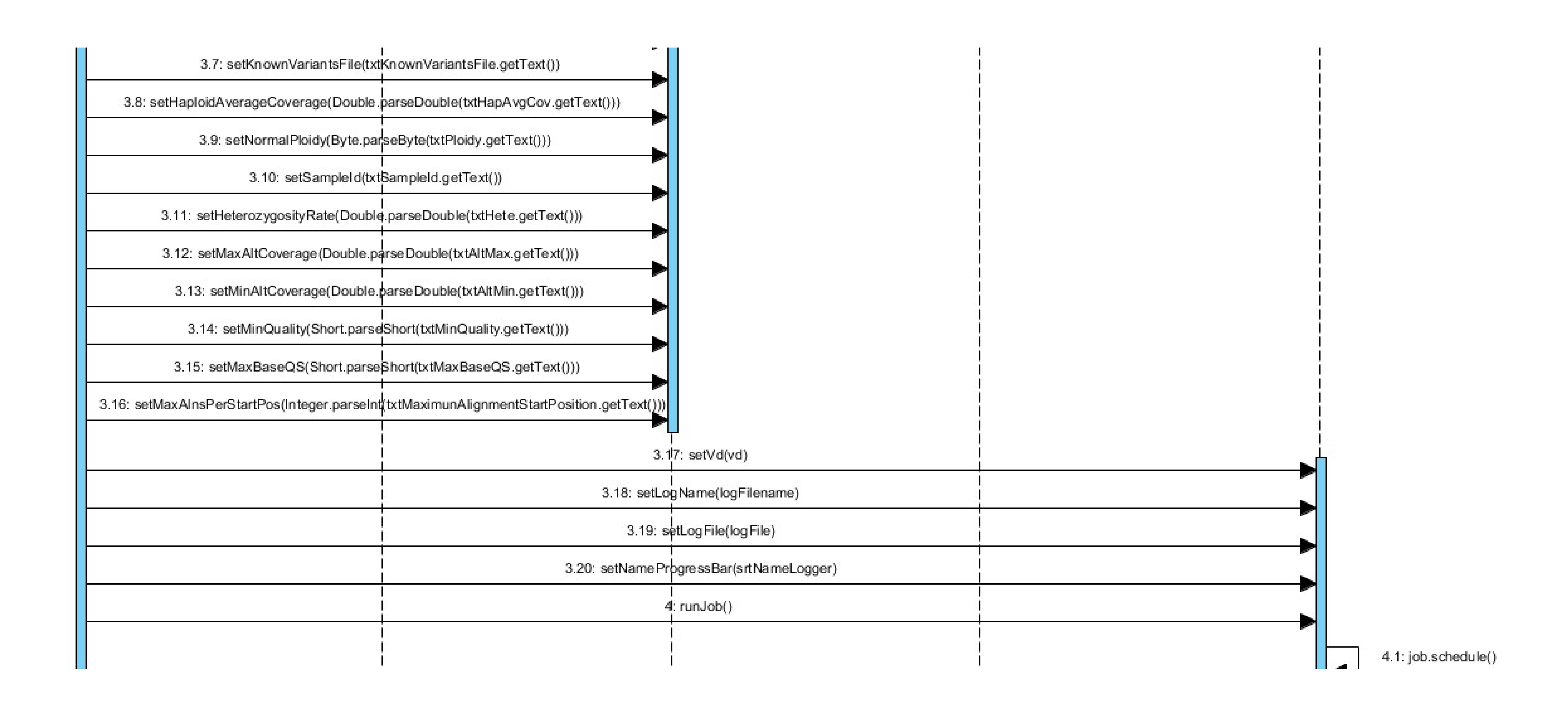
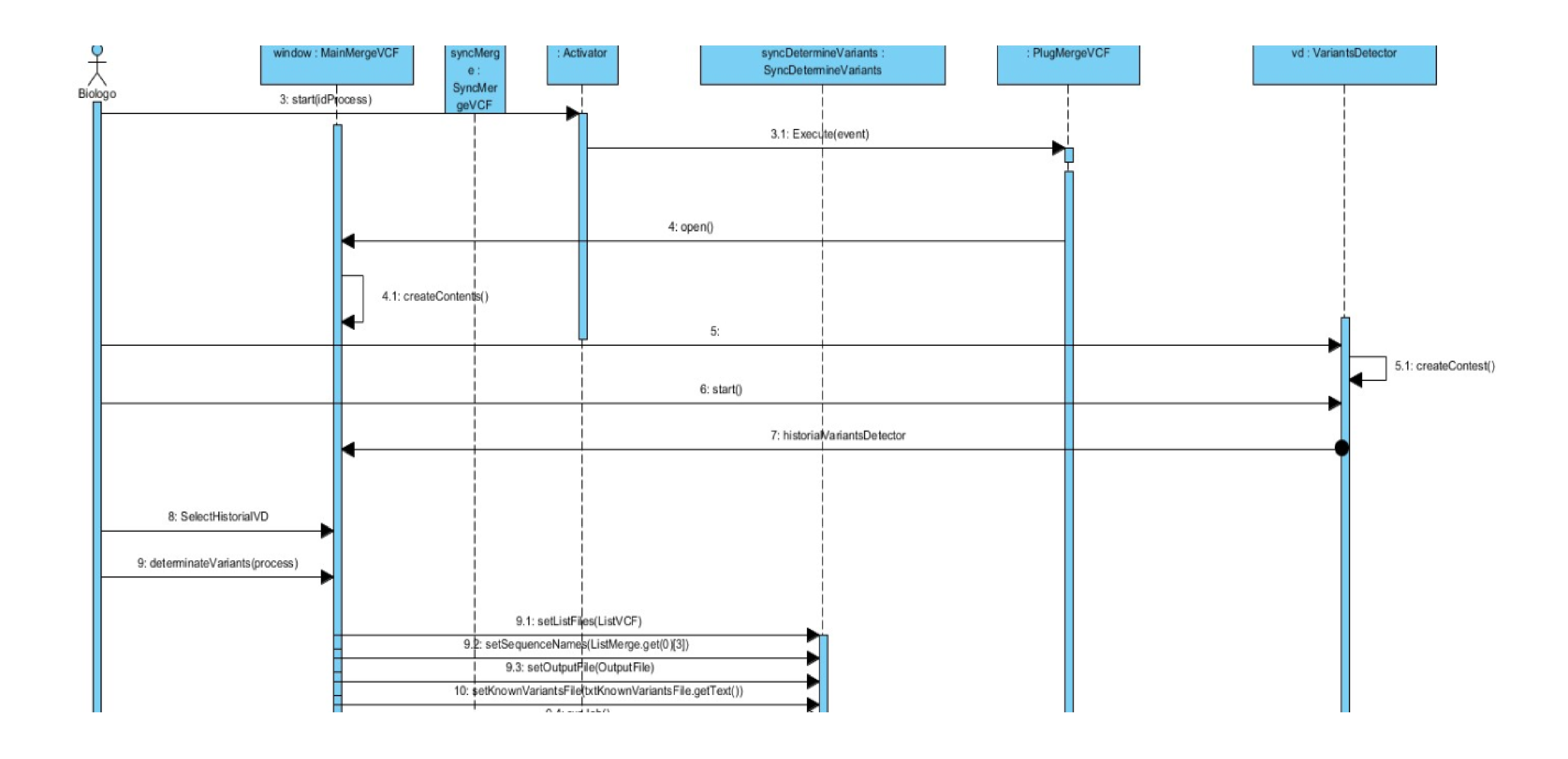


Ilustración 103: Diagrama de secuencia Encontrar Variantes (Este diagrama es una extracción del diagrama original de este caso de uso).



Biologo
window : MainMergeVCF
syncMerge : SyncMergeVCF
: Activator
syncDetermineVariants : SyncDetermineVariants
: PlugMergeVCF
vd : VariantsDetector
3: start(idProcess)
3.1: Execute(event)
4: open()
4.1: createContents()
5:
5.1: createContest()
6: start()
7: historiaVariantsDetector
8: SelectHistorialVD
9: determinateVariants(process)
9.1: setListFiles(ListVCF)
9.2: setSequenceNames(ListMerge.get(0)[3])
9.3: setOutputFile(OutputFile)
10: setKnownVariantsFile(txtKnownVariantsFile.getText())

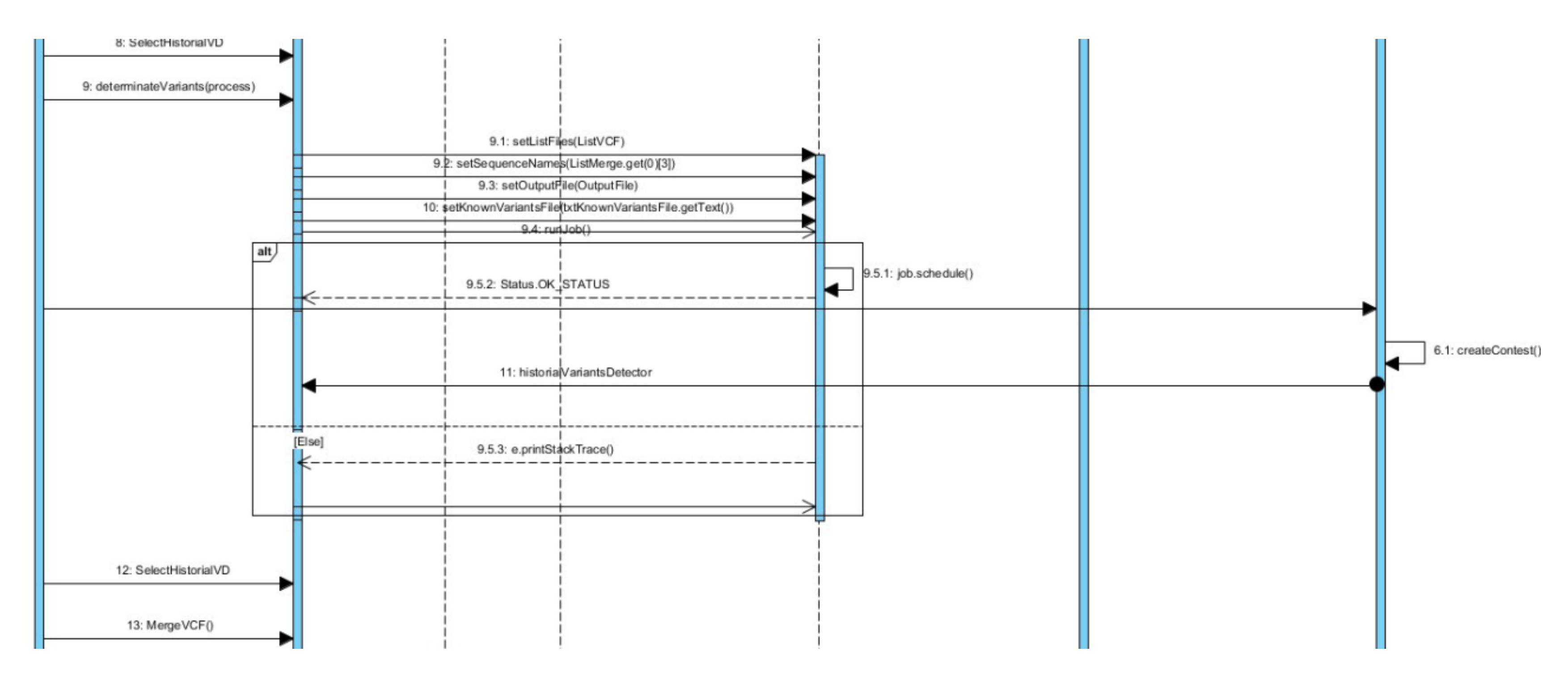


Ilustración 104: Diagrama de Secuencia Mezclar en un solo archivo la información de diferentes muestras analizadas (Este diagrama es una extracción del diagrama original de este caso de uso).

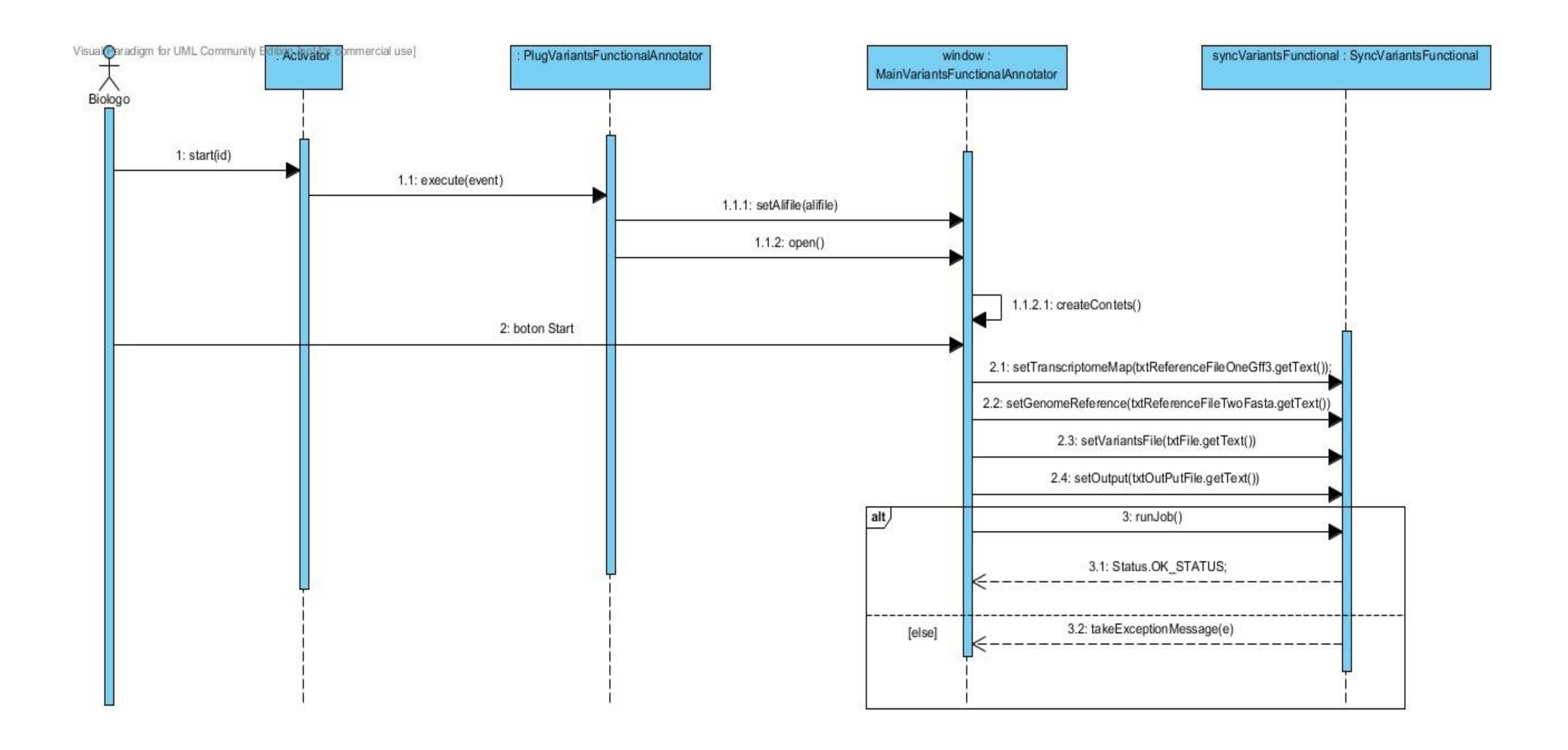
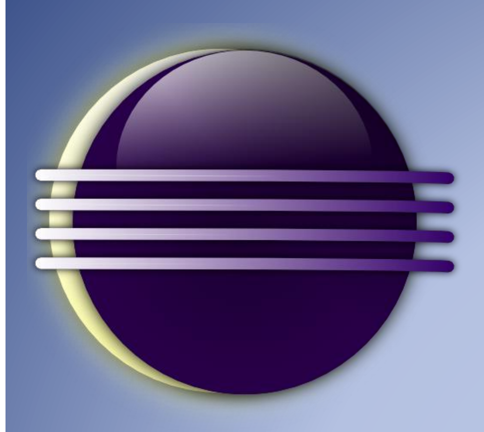


Ilustración 105: Diagrama de secuencia Identificar el efecto de variaciones en los genes.

ANEXO B: Manual de instalación, de seguimiento de NGSEP.





Manual for the Plug-in NGSEP

Daniel Felipe Cruz dfcruz@cgiar.org

Juan Camilo Quintero jcquintero@cgiar.org

Jorge Duitama jduitama@cgiar.org

Contents

Introduction

Next Generation Sequencing (NGS) technologies have increased exponentially the understanding of the genomic structure and function of different organisms within the last decade, including the CIAT mandate crops. In order to handle the vast amount of data produced by these technologies, several bioinformatics tools have been developed to carry on different kinds of analysis. However, most of these tools are not easy to operate, integrate and customize without the technical support of experts in bioinformatics, which produces a bottleneck for several research efforts. This situation poses the need for integrated data analysis pipelines with user friendly interfaces available to the scientific community.

We have developed NGSEP (NGSTools Eclipse Plugin), an integrated framework for variants discovery from NGS data. NGSEP is based on Eclipse which is one of the leading development environments for Java. We integrated previously developed algorithms for SNV detection available in the NGSTools package with Java implementations of state-of-the-art algorithms for CNV and structural variation discovery. NGSEP provides an intuitive interface in which the user has a rich control over the files produced during the different stages of the analysis. These files follow current standard formats such as BAM and VCF, which makes NGSEP results easy to integrate with genome visualization tools. NGSEP can also be integrated with bowtie2 to allow the user to follow all the steps needed to obtain genomic variants from raw reads without scripting. NGSEP will be distributed as an open source project under GPL license to make it available to the scientific community.

System Requirements

In or order to install and execute NGSEP plugin properly you must have installed at least the following components:

-Operative system Windows, Macintosh or Linux.

-Java (jdk 1.6 or higher).

-Eclipse IDE 3.7 or higher. See instructions for how to download and install Eclipse in page 4.

- Bowtie2 is required only for the Map Reads function. See instructions for how to download and install Bowtie2 in the section Map Reads on page 16.

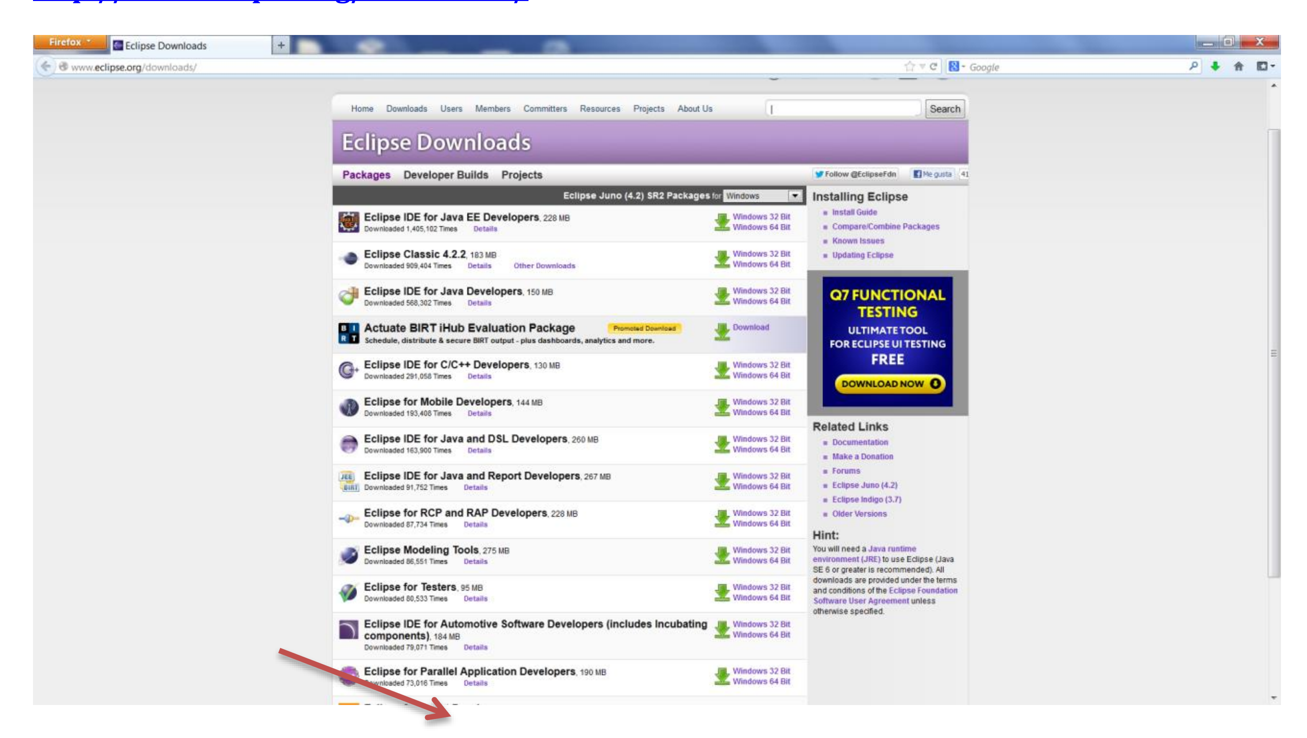
-WinRar or WinZip.

-Text editor. We recommend notepad++. You can downloaded in the following link: http://notepad-plus-plus.org/

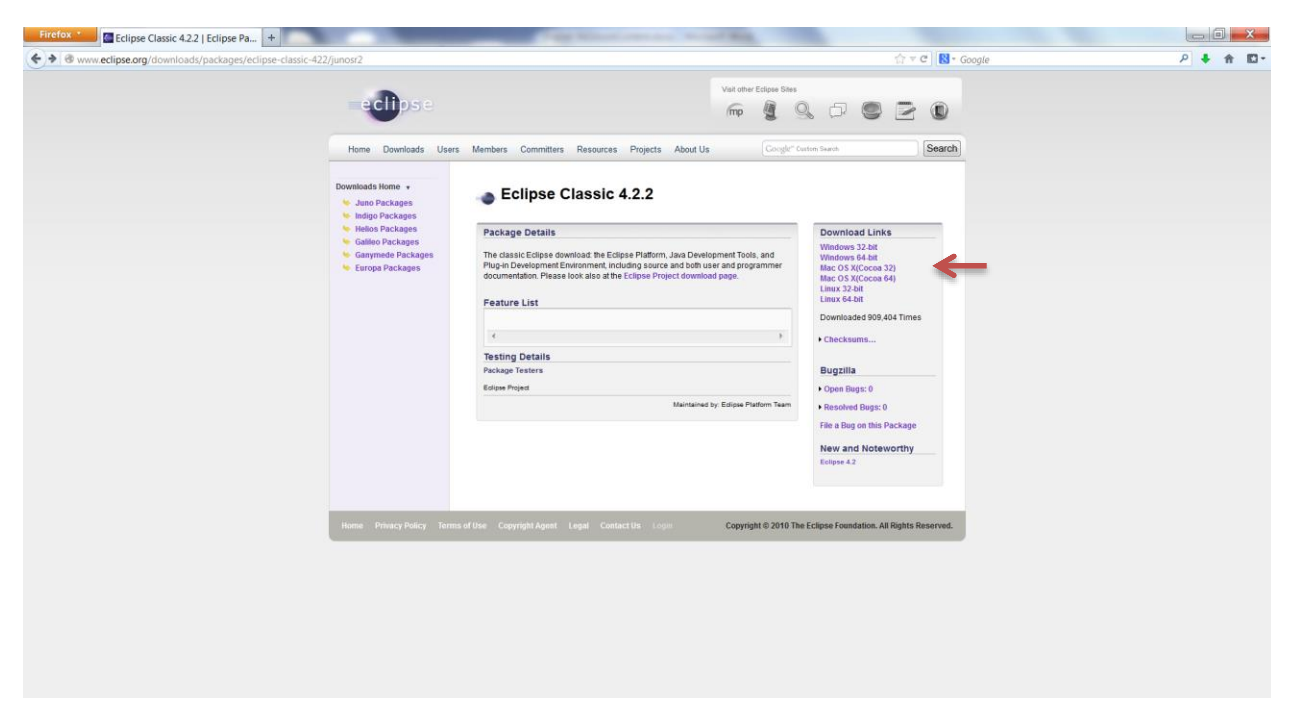
Installing Eclipse IDE

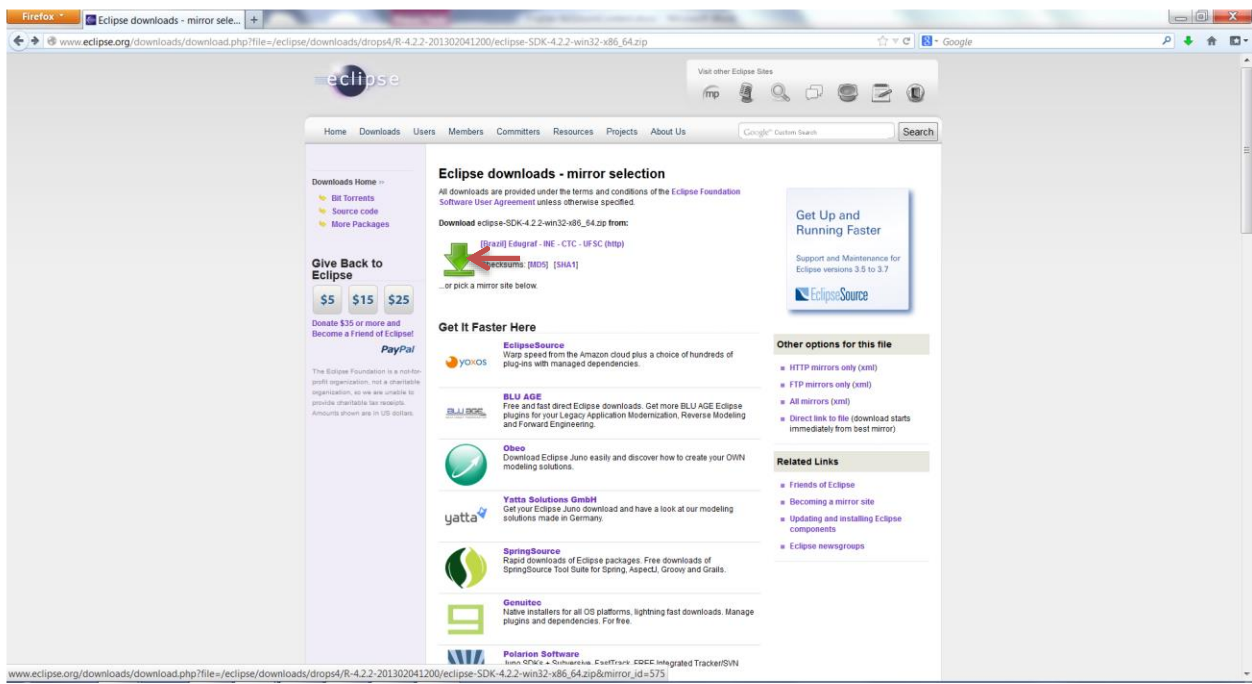
Installing Eclipse Juno 4.2.2.

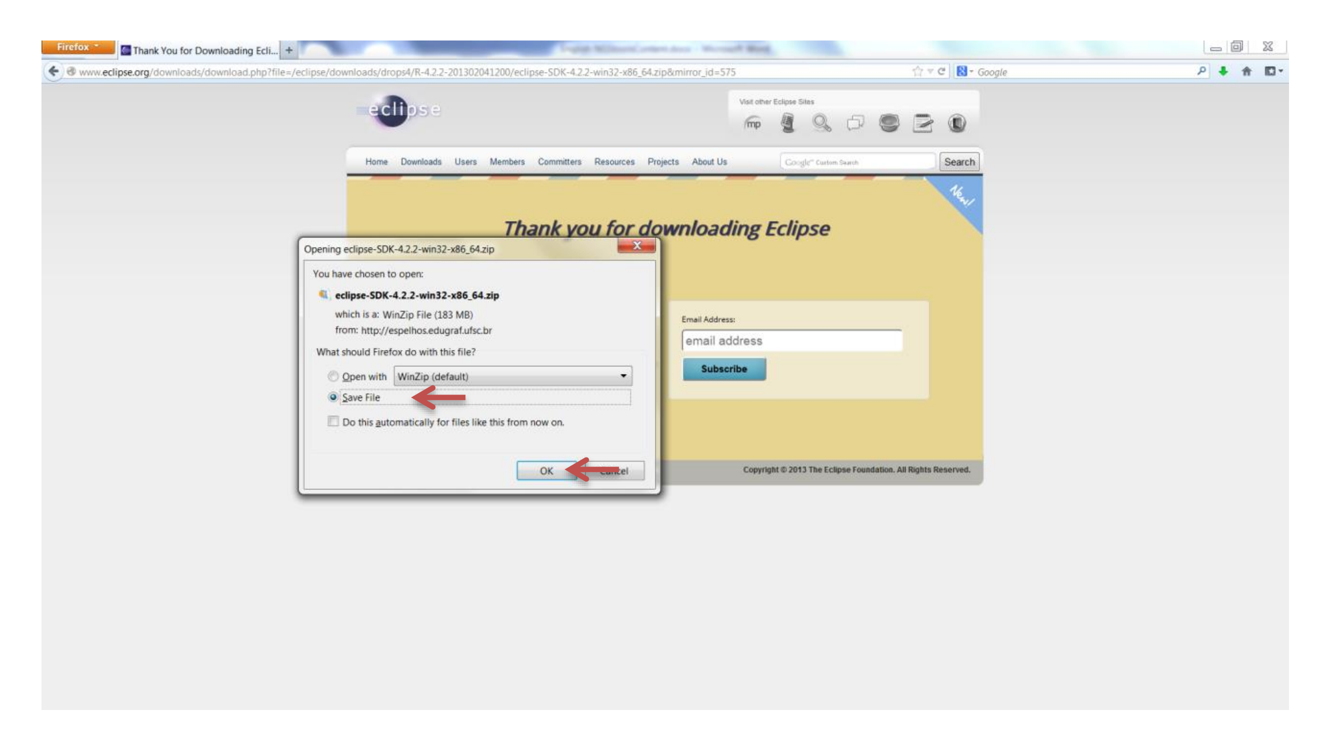
First, download the compressed file from the download page of eclipse organization: http://www.eclipse.org/downloads/ .



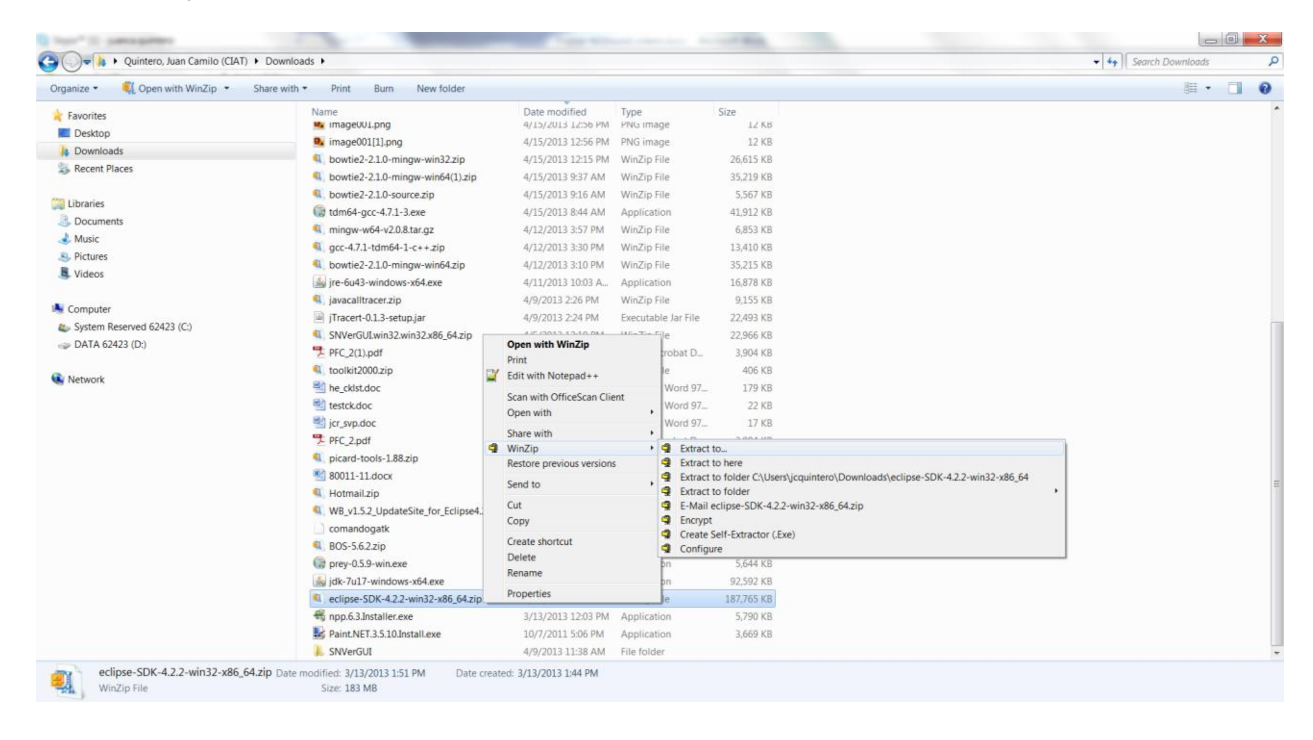
In this page select Eclipse classic and choose the right file according to your operative system and your system architecture (32 or 64 bits).



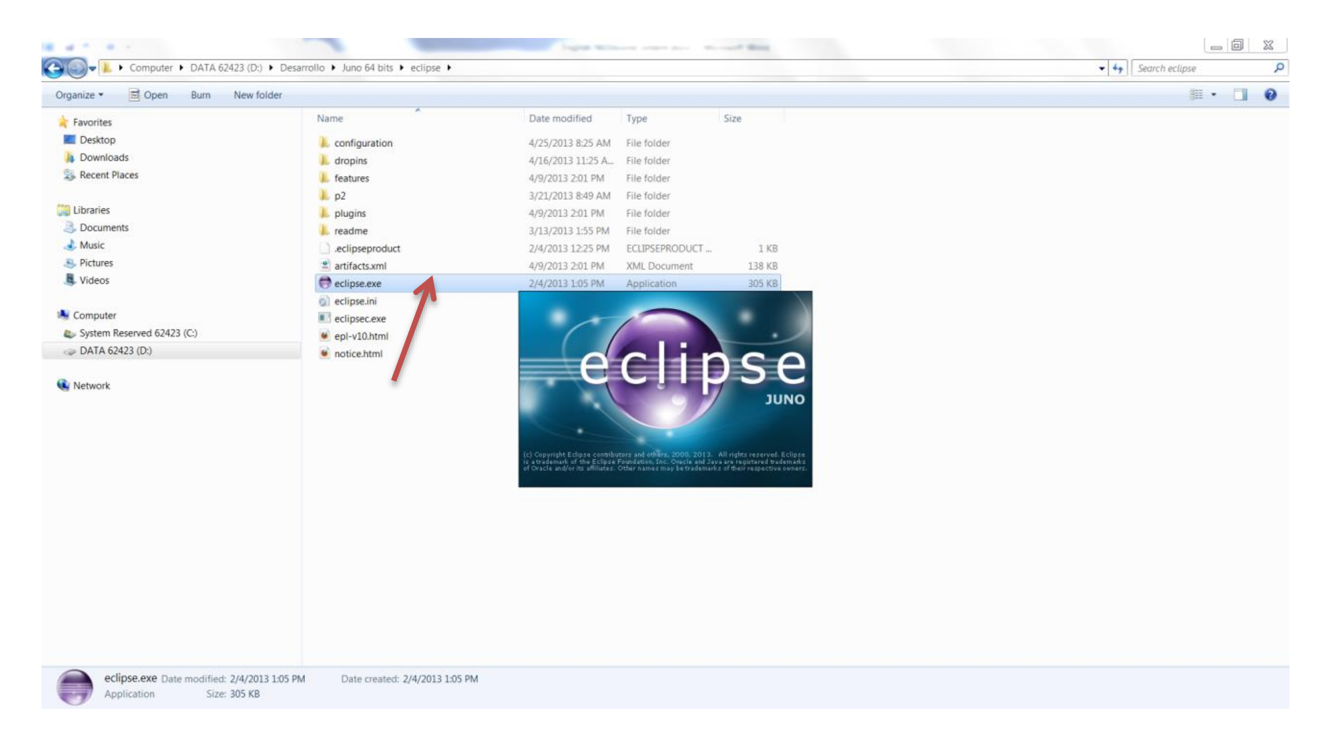




The browser will download automatically the eclipse into a zip file; unzip the file clicking the option extract into your work folder.



When the file is extracted you are going to have a regular folder, in which you will find many files. Your focus should be an executable file called eclipse.exe: Once you click on that file, the eclipse program will be launched and immediately it will ask you for a work folder called workspace. You can select the suggested one or assign a specific one. Now eclipse is ready to be used.



Eclipse will look for your java virtual machine. **If it is not recognized** please follow the next directions:

Once installed, you must edit the PATH variables. In windows you can access them trough: *MY PC – PROPERTIES – ADVANCED OPTIONS*

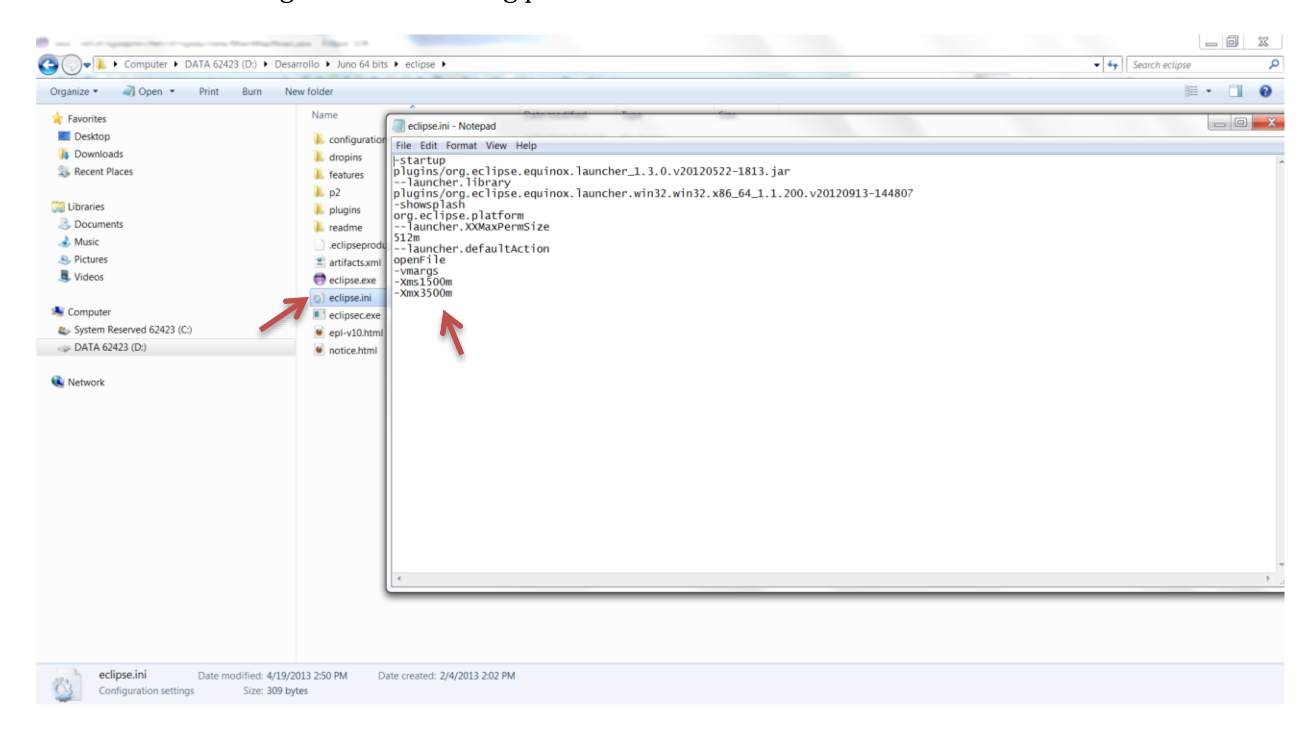
Click on environment variables, search for PATH Variable and edit it adding a " **;** " and the path for the \bin folder from the java folder (where you can find the executable files of eclipse), for example:

"; C:\Program Files\Java\jdk1.6.0_20\bin "

Restart your PC so that the change will be applied, and Java will be available for all the system and therefore for eclipse.

Increasing eclipse memory

It is highly recommended to increase the values of memory granted for eclipse, because NGSEP runs processes which are demanding, producing exceptions in some functionalities when there is not enough memory assigned to eclipse. The most common error that reflects this issue would be **Exception in thread "main" java.lang.OutOfMemoryError: Java heap space.** In order to be able to increase these values of memory, locate eclipse folder and edit a file called **eclipse.ini,** which looks like something like the following picture.



Note: Before editing this file, make sure that eclipse is closed; otherwise the changes will not be applied**.**

Inside that file find the line that says **-Xmx???m**

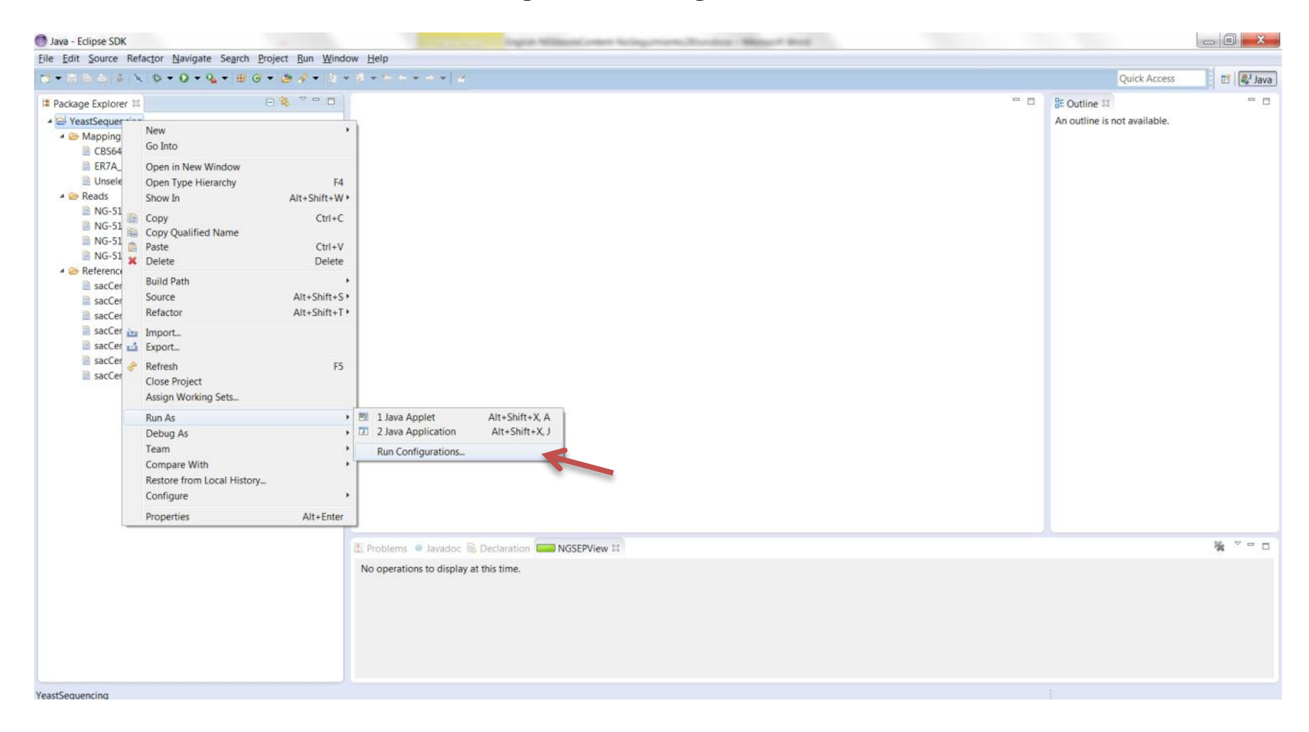
This line indicates how much memory eclipse is allowed to use. In this example we put **Xmx3500m**. It is recommended to use 3 Gb if your RAM memory is higher than 6 Gb otherwise you can try with 1500mb or start decreasing until the eclipse launch successfully.

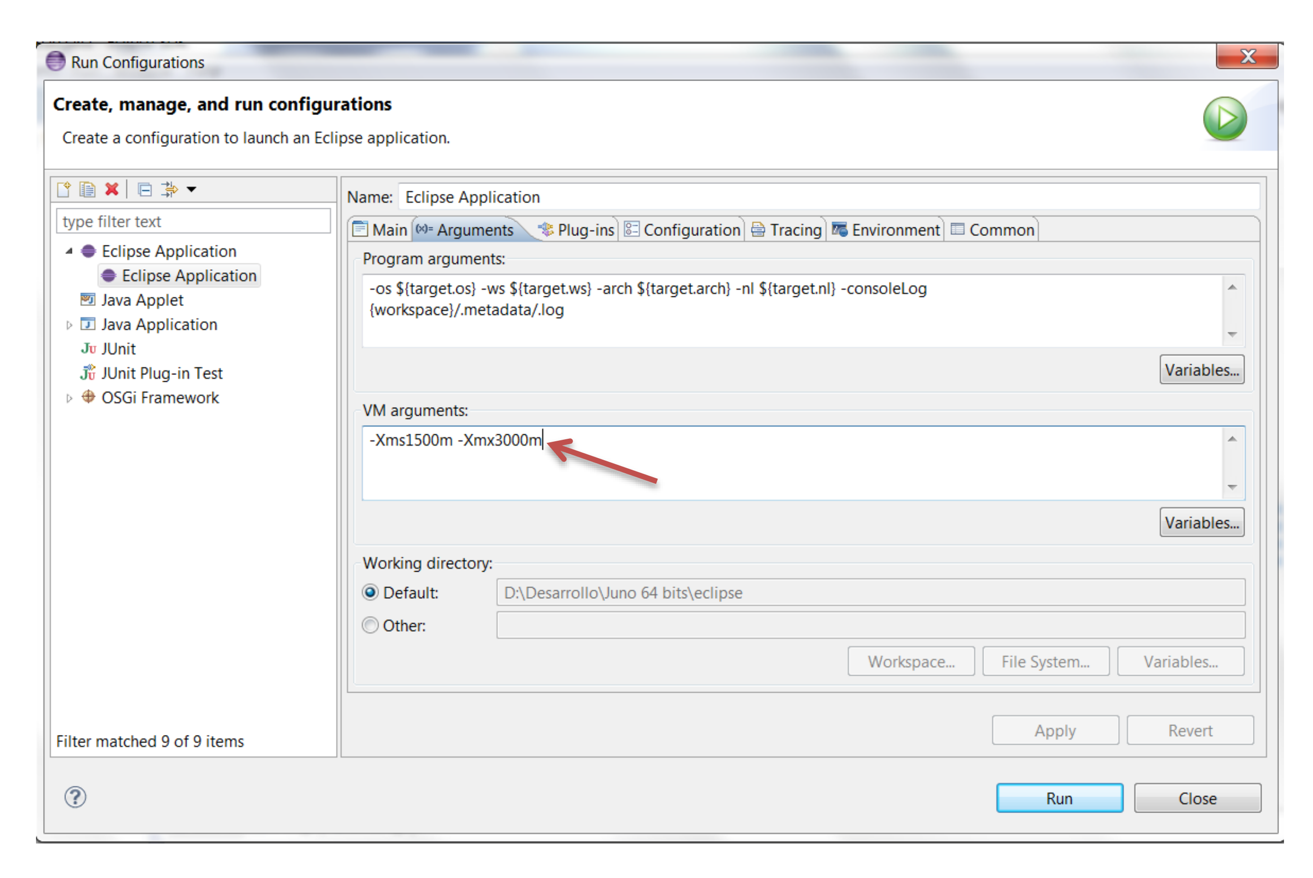
Save and close the file and launch again the eclipse.

Note: The next step is no required if you did the previous one. It only affects the execution of a single application.

Increasing eclipse memory for one application

In order to increase the memory just for the execution of one application, you need to add an additional argument. This can be added making a right click in the application you want to run, and then in the menu: Run As → Run Configuration→ Arguments, as follows:





In the option VM arguments, indicate how much memory you want to dedicate. The values should indicate the minimum and maximum memory allowed. For example:

- Xms1500m

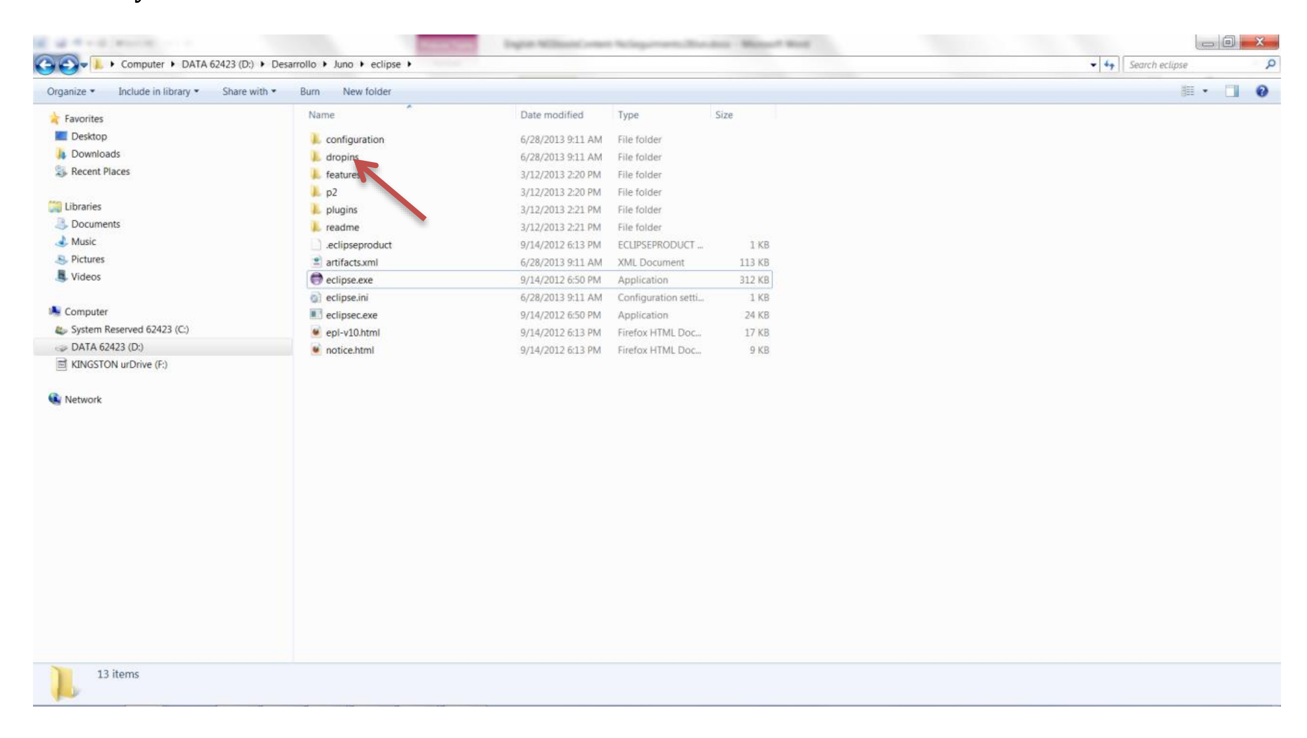
- Xmx3000m

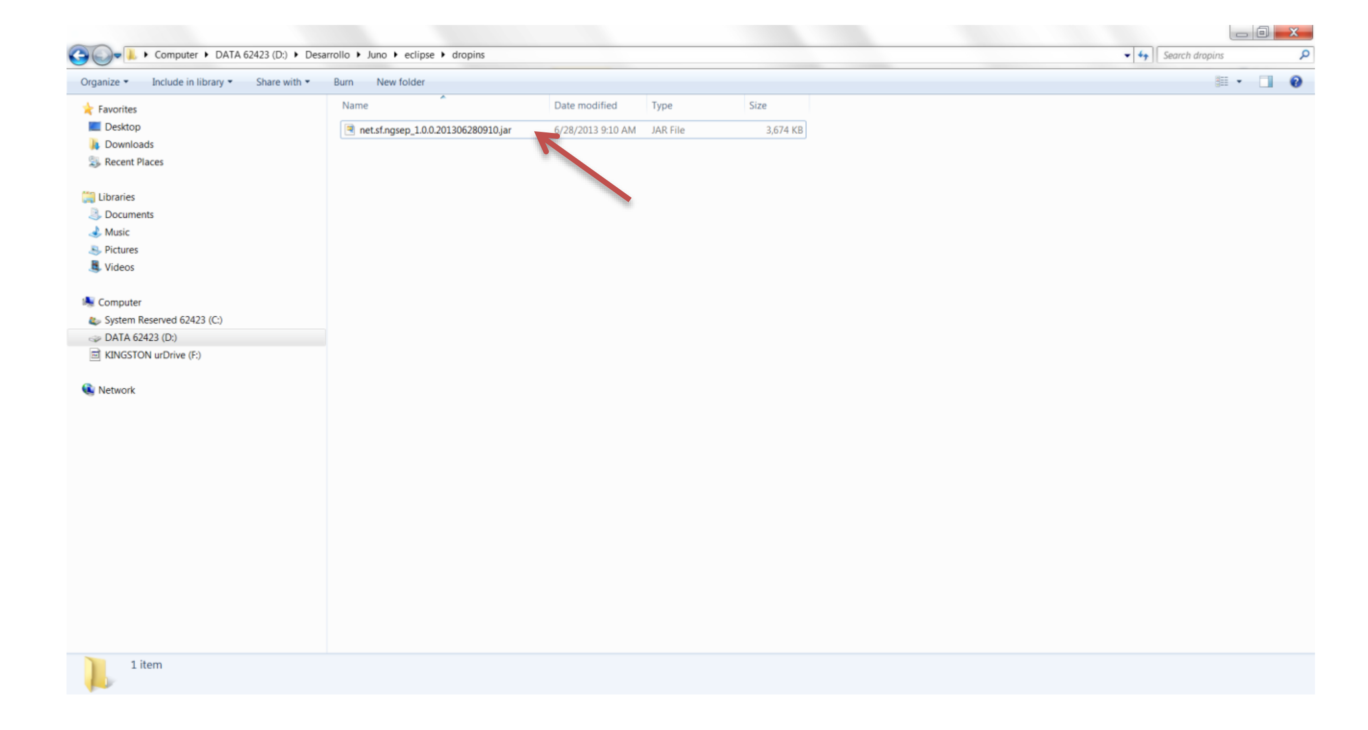
This means that you want to grant 1500 Mb as minimum and 3000 Mb or 3 Gb at top.

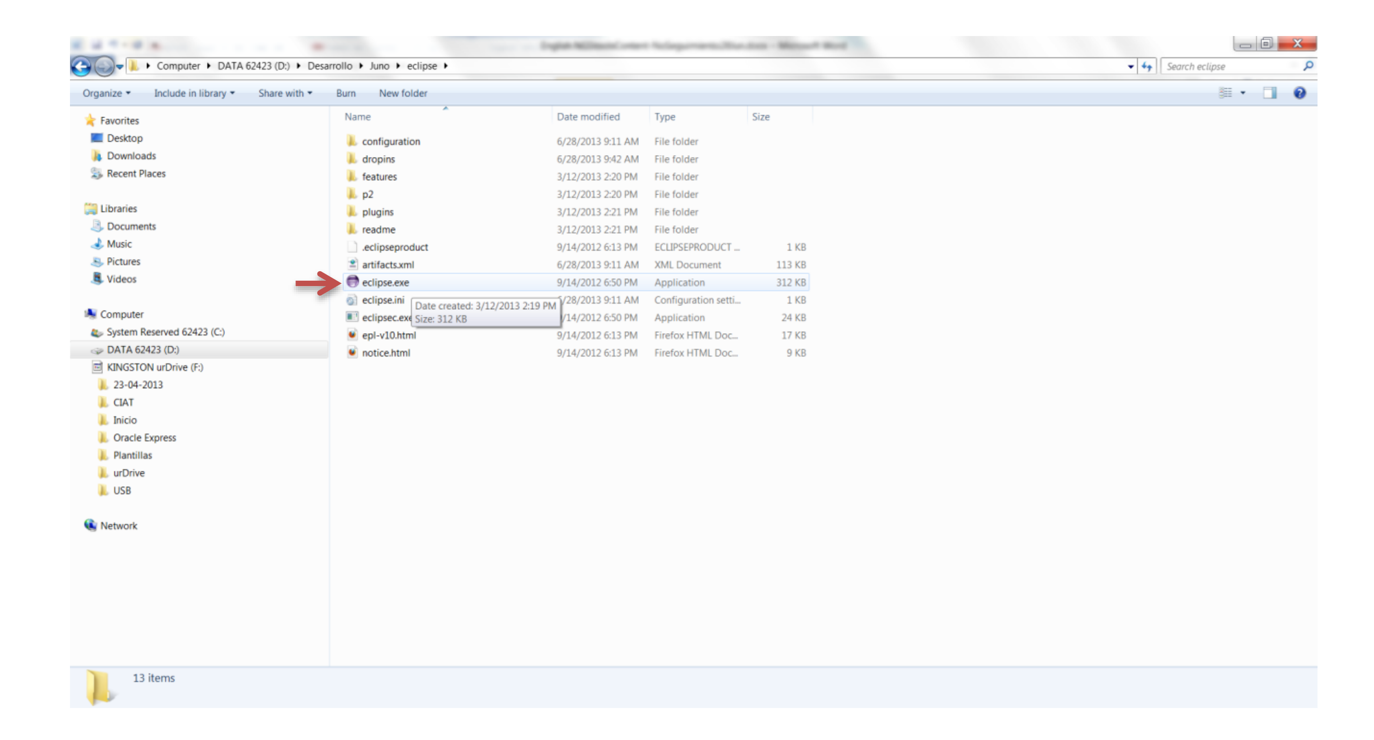
Save and close the file and launch again the eclipse.

NGSEP plugin installation

After downloading the NGSEP jar file, you need to paste it in the dropins folder in the eclipse directory



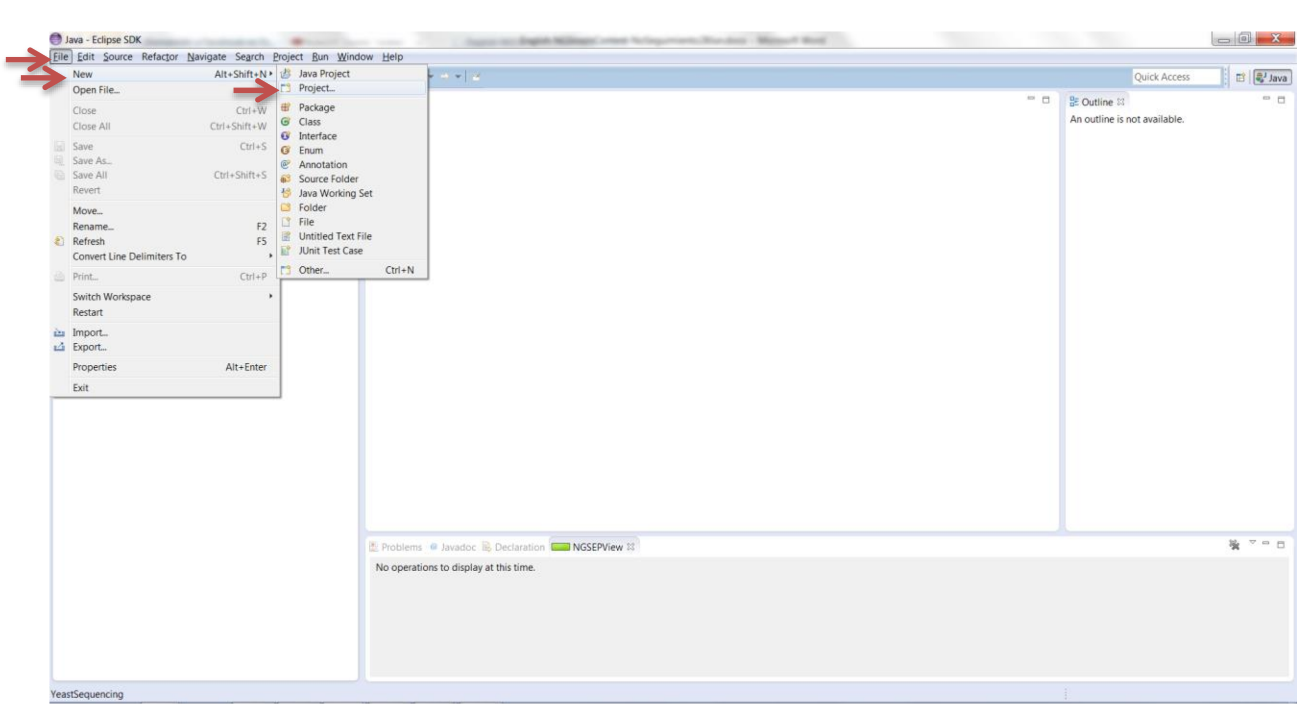




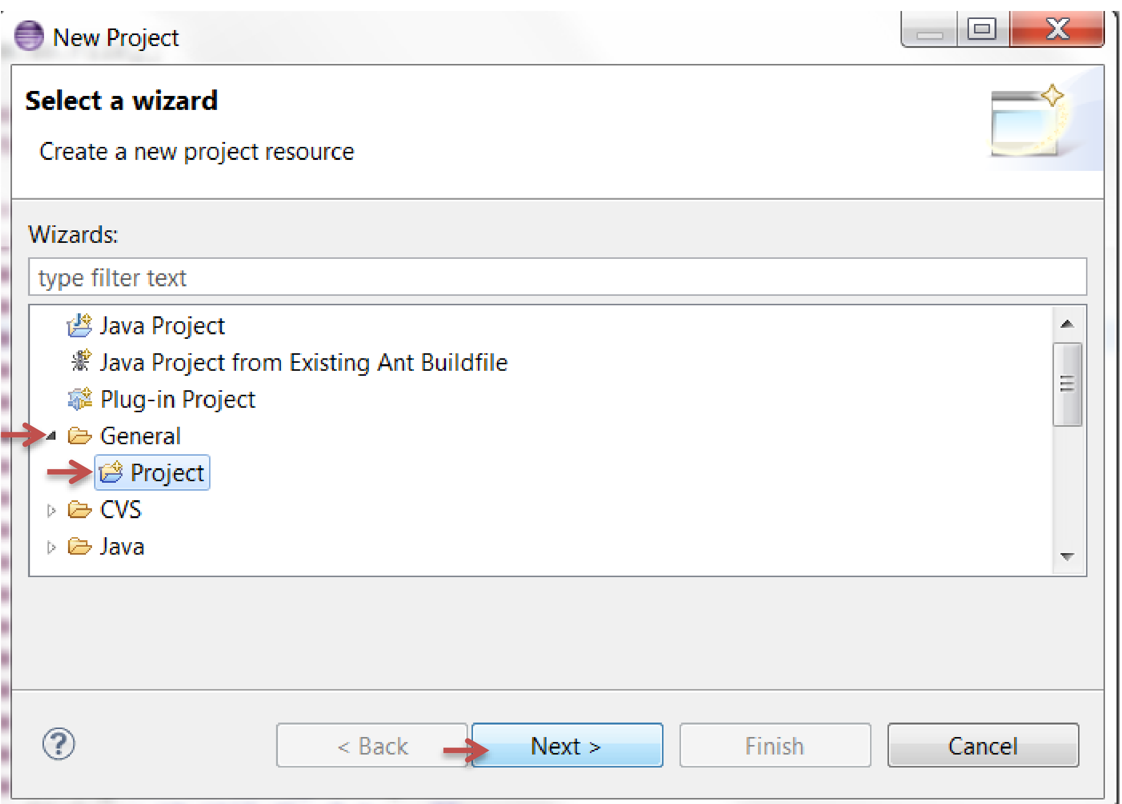
You need to start eclipse again and the NGSEP will be integrated with eclipse IDE.

Using NGSEP plugin

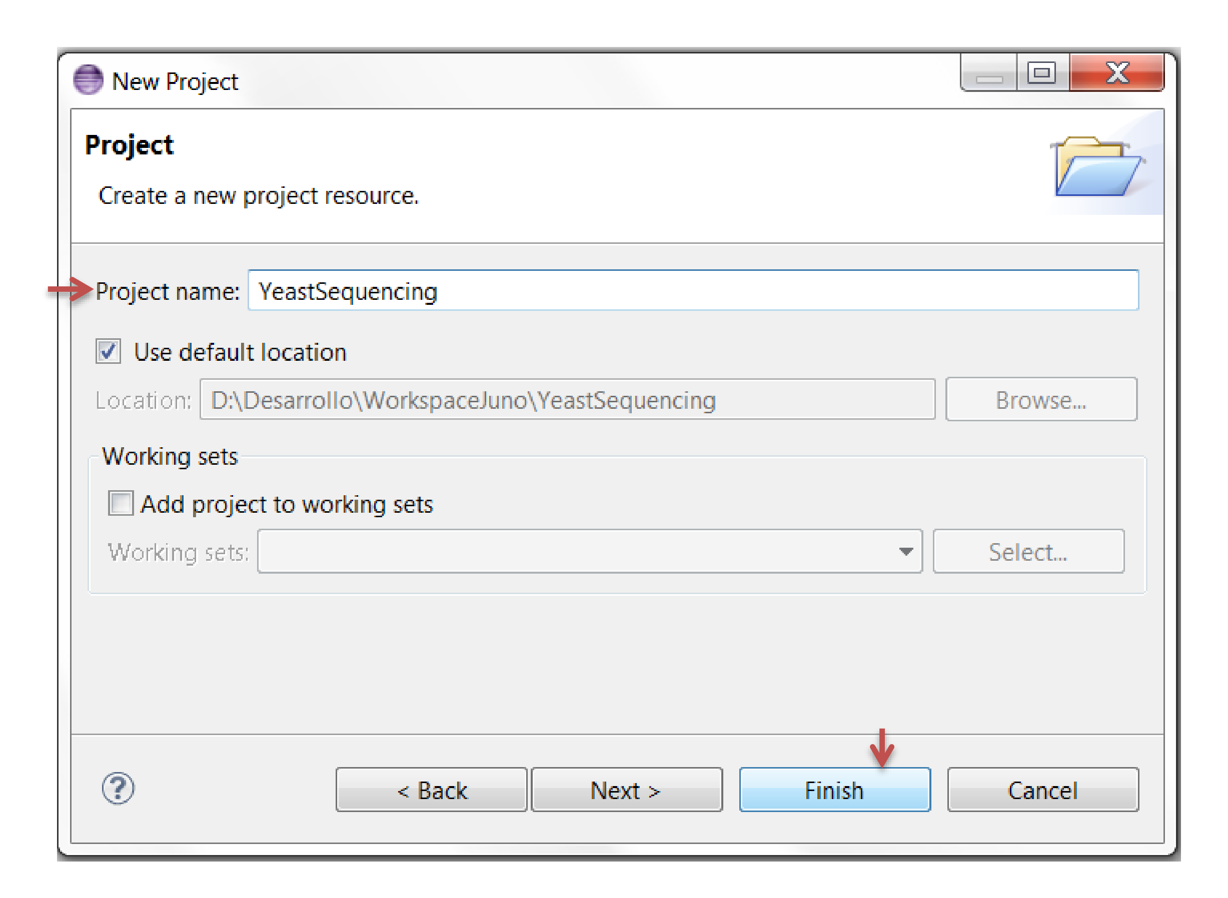
The first thing that you need to do after starting eclipse is to create a new project. To do so, go to the task bar at the upper part of eclipse, and select: File →New Project, and choose General → Project. Immediately a window to name the project will show up, where you can type the name of your new project..



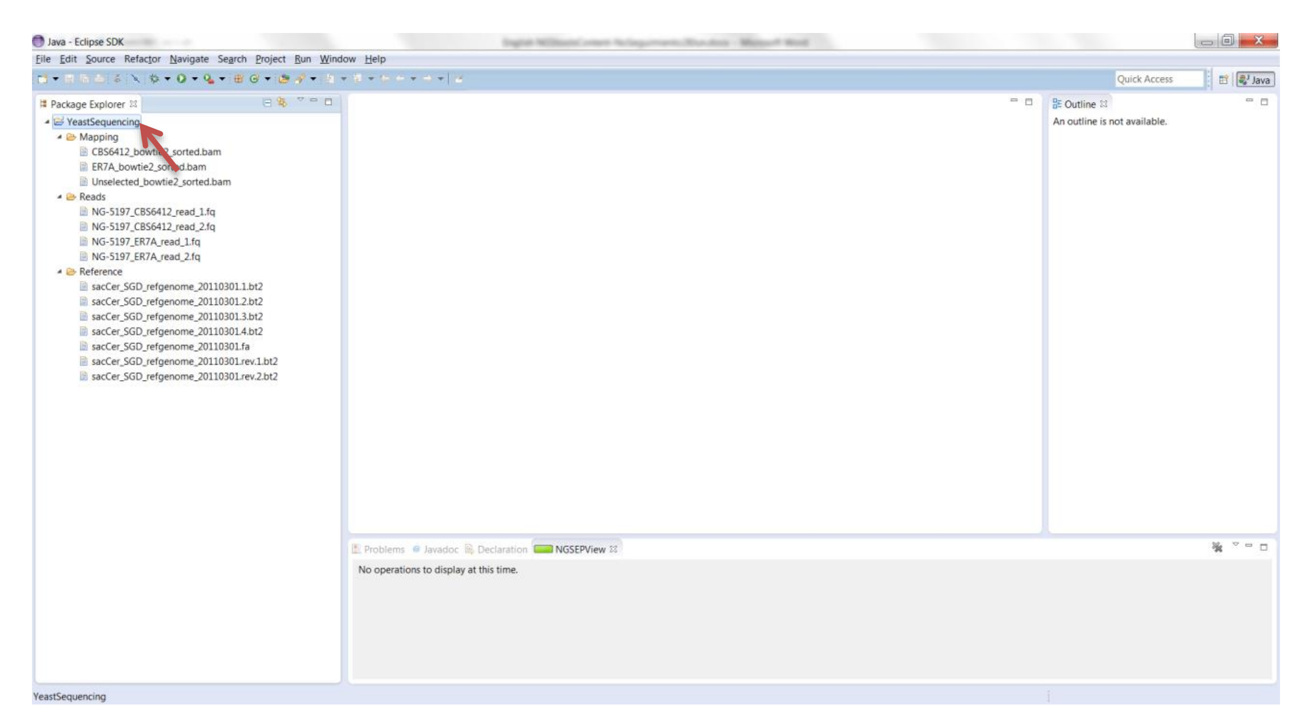
Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
New Alt+Shift+N
Open File...
Close Ctrl+W
Close All Ctrl+Shift+W
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
Revert
Move...
Rename... F2
Refresh F5
Convert Line Delimiters To
Print... Ctrl+P
Switch Workspace
Restart
Import...
Export...
Properties Alt+Enter
Exit
Java Project
Project...
Package
Class
Interface
Enum
Annotation
Source Folder
Java Working Set
Folder
File
Untitled Text File
JUnit Test Case
Other... Ctrl+N
Quick Access
Java
Outline
An outline is not available.
Problems
Javadoc
Declaration
NGSEPView
No operations to display at this time.
YeastSequencing



New Project
Select a wizard
Create a new project resource
Wizards:
type filter text
Java Project
Java Project from Existing Ant Buildfile
Plug-in Project
General
Project
CVS
Java
< Back
Next >
Finish
Cancel



Now you can add your input files to the new project. The input files could be BAM, SAM and Fasta formats:



Now NGSEP should be working in your eclipse. If you do right click on any input file, for example the .Bam, you will see several options, and you should be able to see NGSEP among them. If you put the mouse cursor on it you will see the bioinformatics options that NGSEP can execute, from map reads to variants detection and statistics plots.

Enable NGSEP Progress Bar

Enabling the NGSEP view in eclipse, will allow you to see the progress bar of the NGSEP tasks.

In order to enable the progress bar go to the task bar at the upper part of the IDE and select the following options:

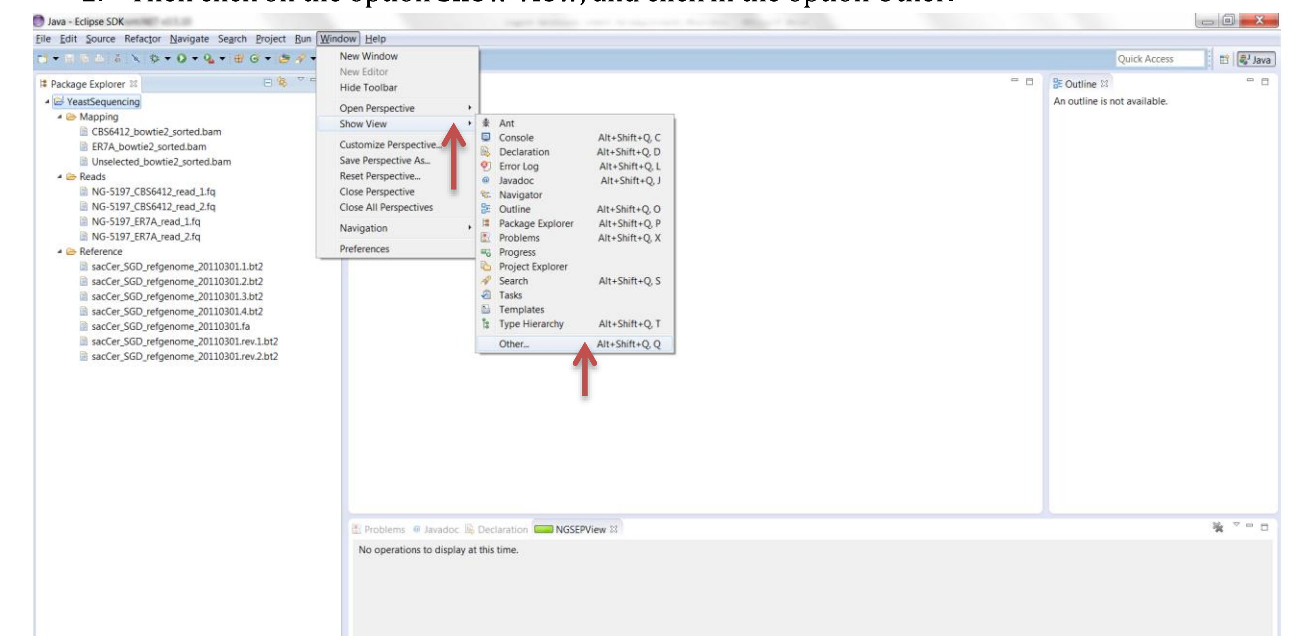
Windows → Show view→Other→NGSEPView

 **Note**: you could find some differences among Eclipse versions.

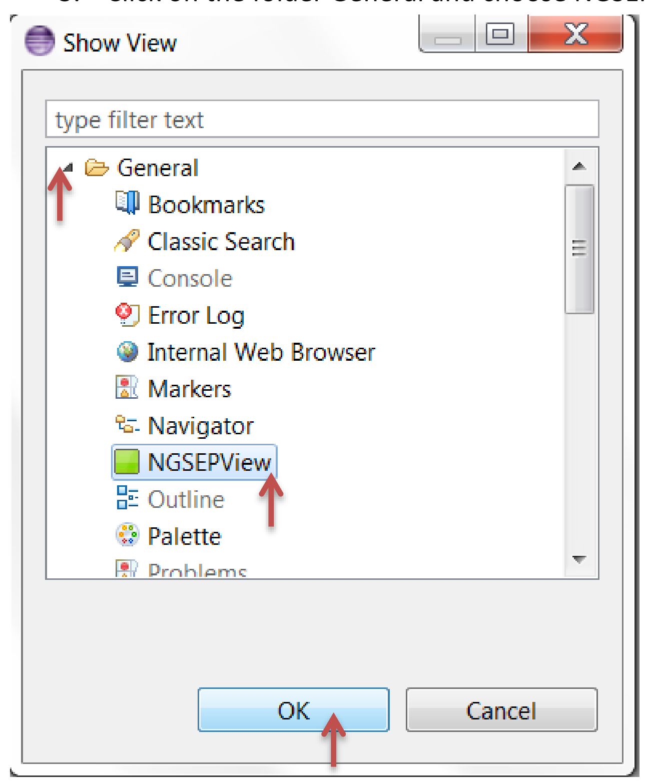
Remember, you have to paste the plugin in the droprins folder; otherwise the progress bar will not be displayed.

1. First click on window option in the task bar.

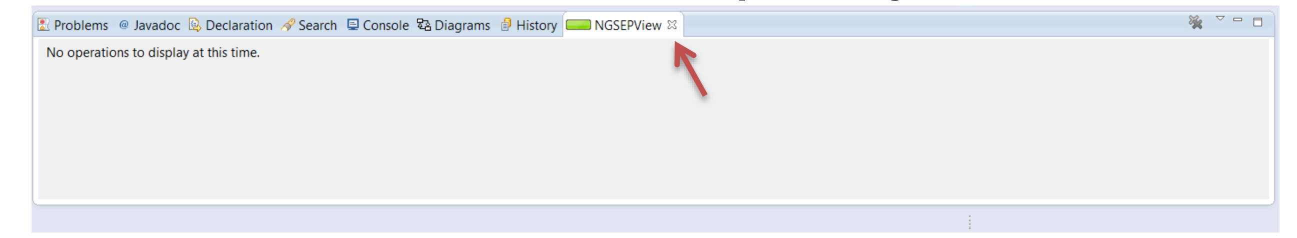
2. Then click on the option **Show view**, and click in the option Other**.**



3. Click on the folder General and choose NGSEPView



4. You will be able to see a new tab next to the console and problems log.



Note: This tab contains the progress bars of NGSEP. If you haven't triggered any process you should not see anything there, however sometimes eclipse uses that tab to report processes of projects and its environment. Do not worry if that happens. Now with the progress bar view activated you are ready to use the different options that NGSEP offers.

Map Reads

This process executes the matching between a reference genome and reads that come from sequencers such as Illumina and 454.

REQUIREMENTS

- Bowtie2: Open source tool that is able to map up to 25 million of short reads (35 pb) per hour.

The first step to use Map reads is downloading and installing bowtie2 in your PC. You can download bowtie2 in the following link: http://sourceforge.net/projects/bowtie-bio/files/bowtie2/2.1.0/

For installing bowtie2 in windows operating system follow these steps:

1. Download Bowtie2 and extract the code to a location on your disk.
2. Find the executable: bowtie2-align.exe, bowtie2-build.exe.
3. Add bowtie2 to your PATH environment variable. To do this, follow your operation system's instructions for adding the directory to your Path. For Windows follow these steps:

    - Make a right click on Computer and choose Properties → Advanced System Settings → Advanced Options → System Variables →Path → Edit. In the option Variable Value, add a ";" (each semicolon adds a new path for variables) and write the path where your bowtie2 folder is.
      For example:
      ;C:\Users\jcquintero\Desktop\CIAT\Bowtie2\bowtie2-2.1.0\bowtie2-align.exe.

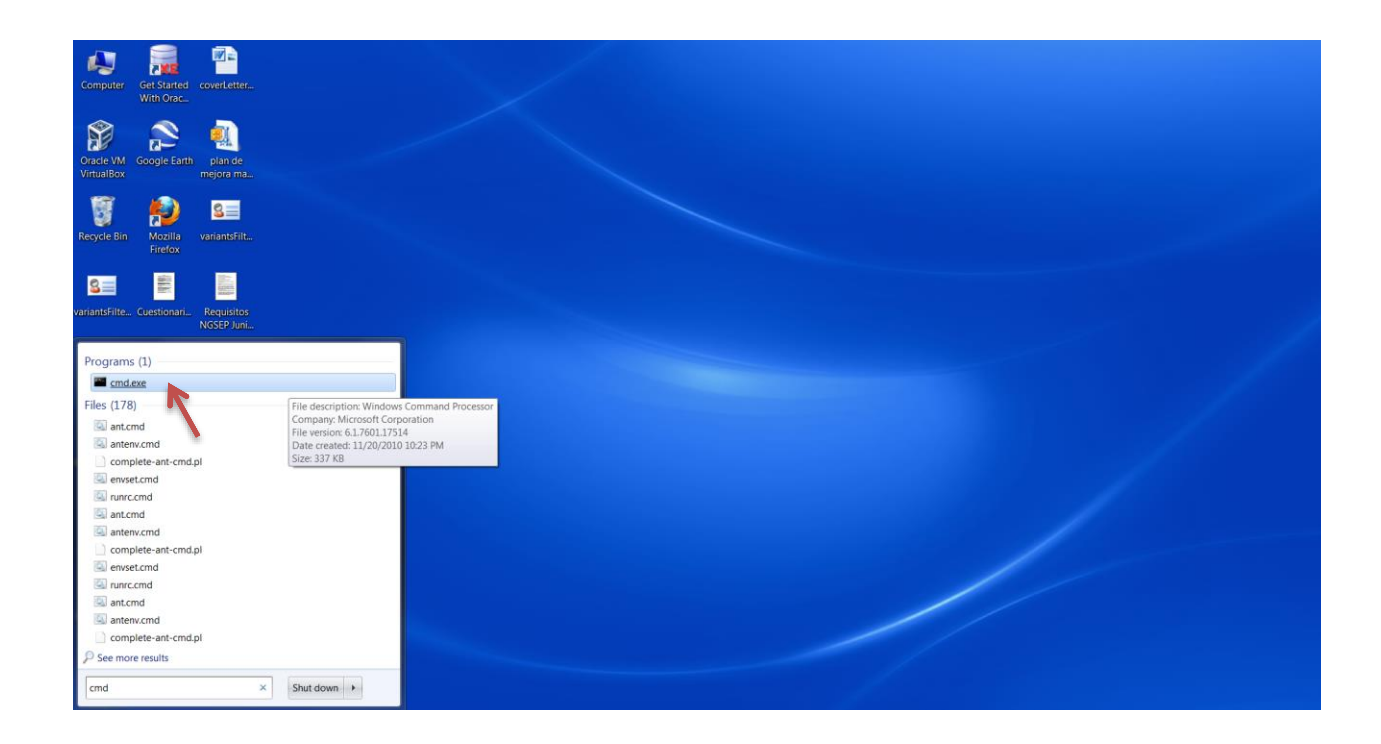
To perform the Map reads function you will also need the reference genome indexed by bowtie2 order for faster execution; otherwise the process will not start.

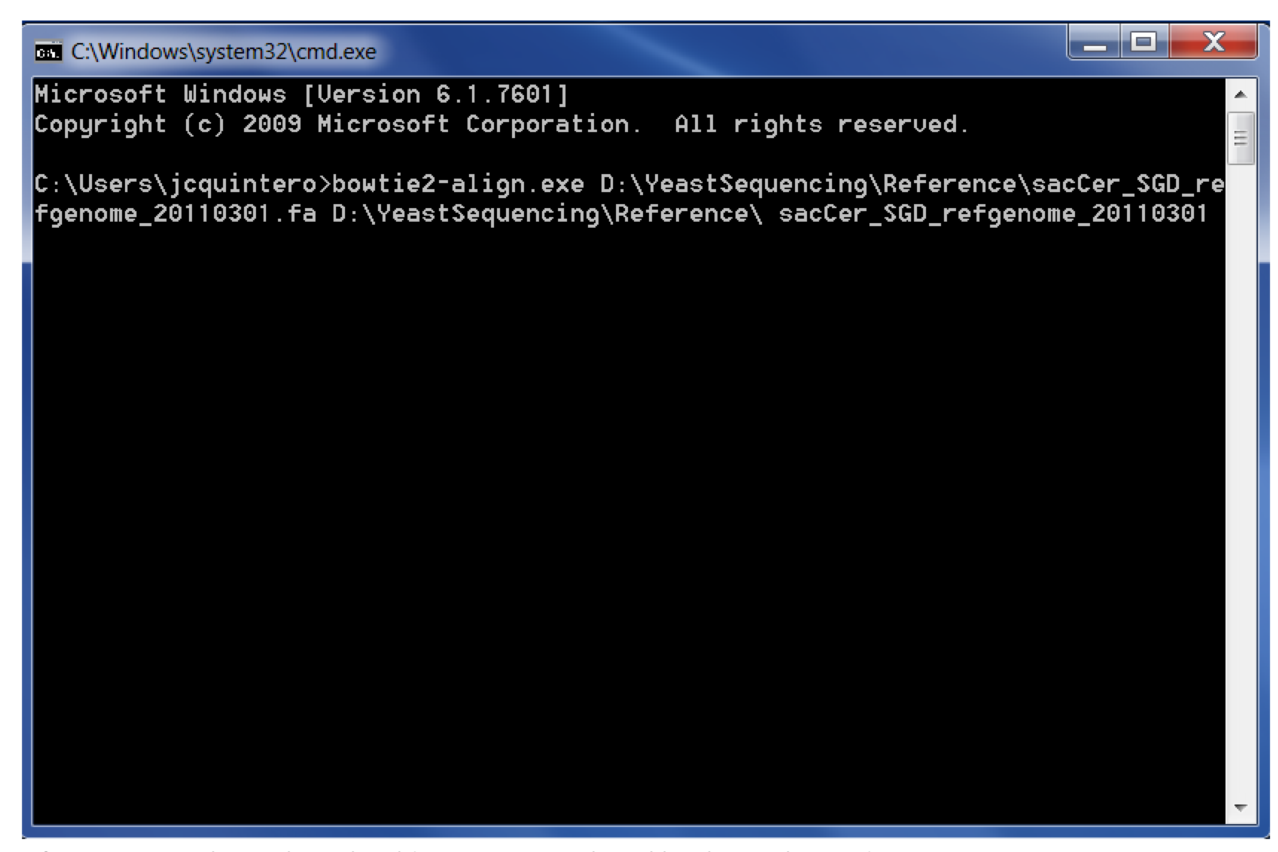
You can index the reference genome in bowtie2 for Windows as follows:

- Call the command line by typing cmd in the Windows search bar
- Indicate in the cmd the path where you have the reference genome, and the path and new name for the indexed reference file that you are creating.

For example:

bowtie2-build.exe
D:\YeastSequencing\Reference\sacCer_SGD_refgenome_20110301.fa
D:\YeastSequencing\Reference\ sacCer_SGD_refgenome_20110301





After you type the path, it should appear something like this in the cmd:

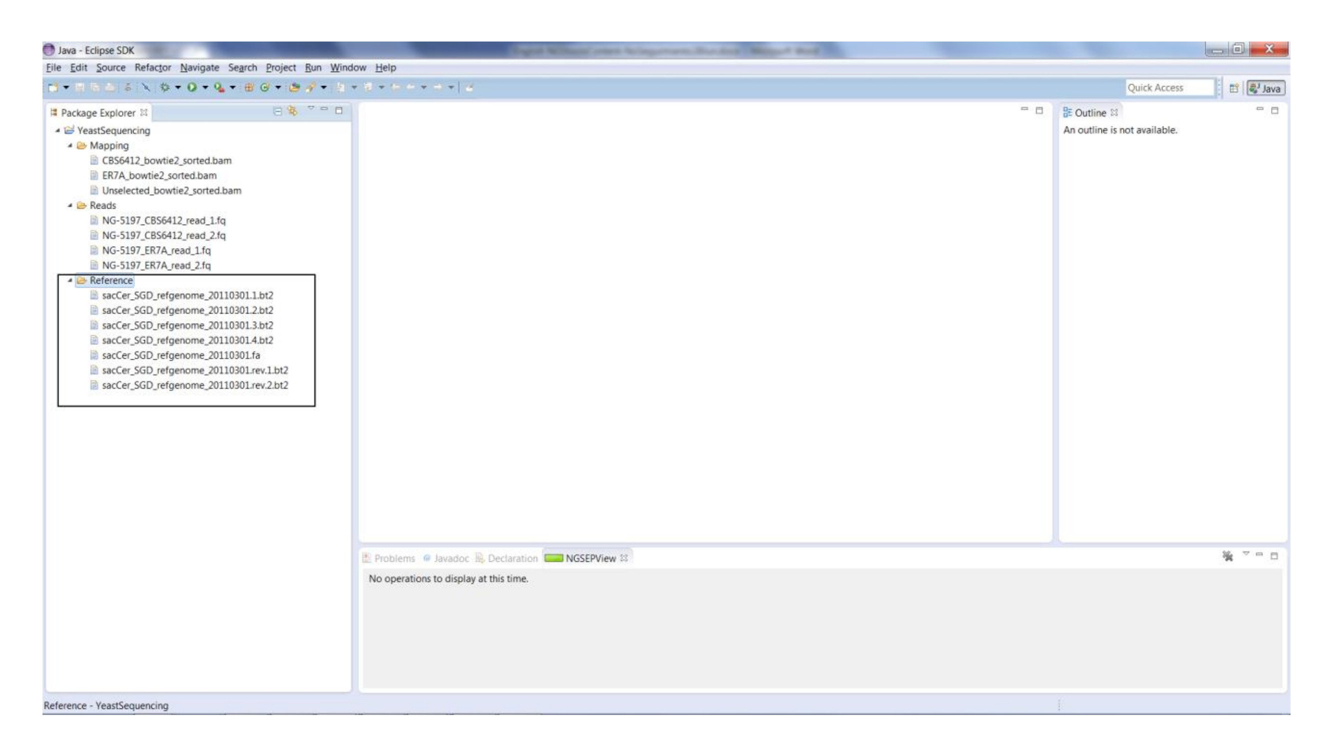
C:\Windows\system32\cmd.exe

```
  90%
  100%
  Block accumulator loop time: 00:00:00
  Sorting block of length 2059905
  (Using difference cover)
  Sorting block time: 00:00:01
Returning block of 2059906
Exited Ebwt loop
fchr[A]: 0
fchr[C]: 3766349
fchr[G]: 6086925
fchr[T]: 8404025
fchr[$]: 12157105
Exiting Ebwt::buildToDisk()
Returning from initFromVector
Wrote 8247244 bytes to primary EBWT file: D:\Desarrollo\sacCer_SGD_refgenome_201
10301.rev.1.bt2
Wrote 3039284 bytes to secondary EBWT file: D:\Desarrollo\sacCer_SGD_refgenome_2
0110301.rev.2.bt2
Re-opening _in1 and _in2 as input streams
Returning from Ebwt constructor
Headers:
    len: 12157105
    bwtLen: 12157106
    sz: 3039277
    bwtSz: 3039277
    lineRate: 6
    offRate: 4
    offMask: 0xfffffff0
    ftabChars: 10
    eftabLen: 20
    eftabSz: 80
    ftabLen: 1048577
    ftabSz: 4194308
    offsLen: 759820
    offsSz: 3039280
    lineSz: 64
    sideSz: 64
    sideBwtSz: 48
    sideBwtLen: 192
    numSides: 63319
    numLines: 63319
    ebwtTotLen: 4052416
    ebwtTotSz: 4052416
    color: 0
    reverse: 1
Total time for backward call to driver() for mirror index: 00:00:08

C:\Users\jcquintero>_
```

At the end of the indexing process, you should have the following files in the path that you previously indicated for your output:

sacCer_SGD_refgenome_20110301.1.bt2	5/7/2013 11:53 AM	BT2 File	8,054 KB
sacCer_SGD_refgenome_20110301.2.bt2	5/7/2013 11:53 AM	BT2 File	2,969 KB
sacCer_SGD_refgenome_20110301.3.bt2	5/7/2013 11:53 AM	BT2 File	1 KB
sacCer_SGD_refgenome_20110301.4.bt2	5/7/2013 11:53 AM	BT2 File	2,969 KB
sacCer_SGD_refgenome_20110301.rev.1.b...	5/7/2013 11:53 AM	BT2 File	8,054 KB
sacCer_SGD_refgenome_20110301.rev.2.b...	5/7/2013 11:53 AM	BT2 File	2,969 KB

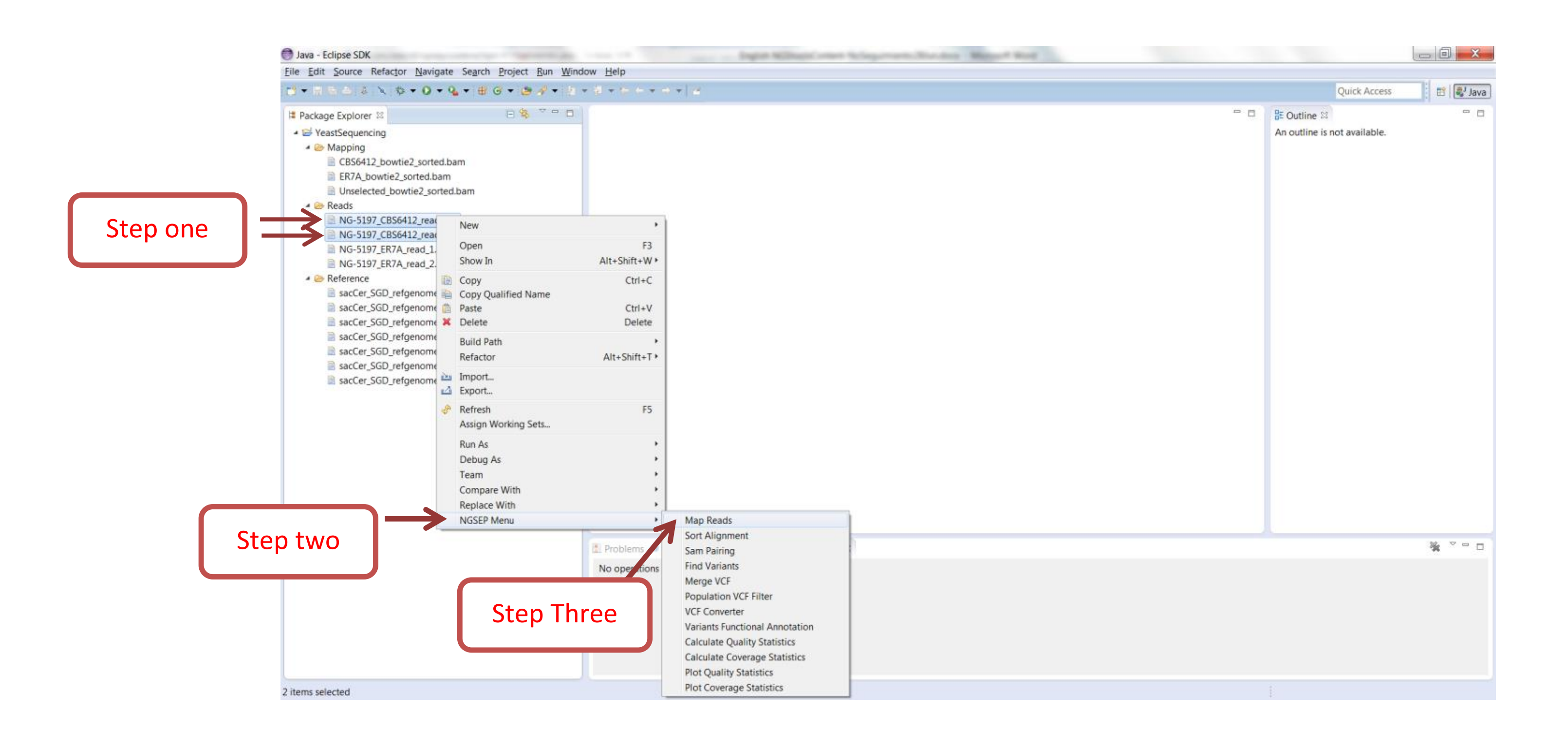


For more information about the indexing process in bowtie2, we recommend these links:
http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml
http://sauron.cs.umd.edu/bowtie2/doc/manual.html - the-bowtie2-build-indexer

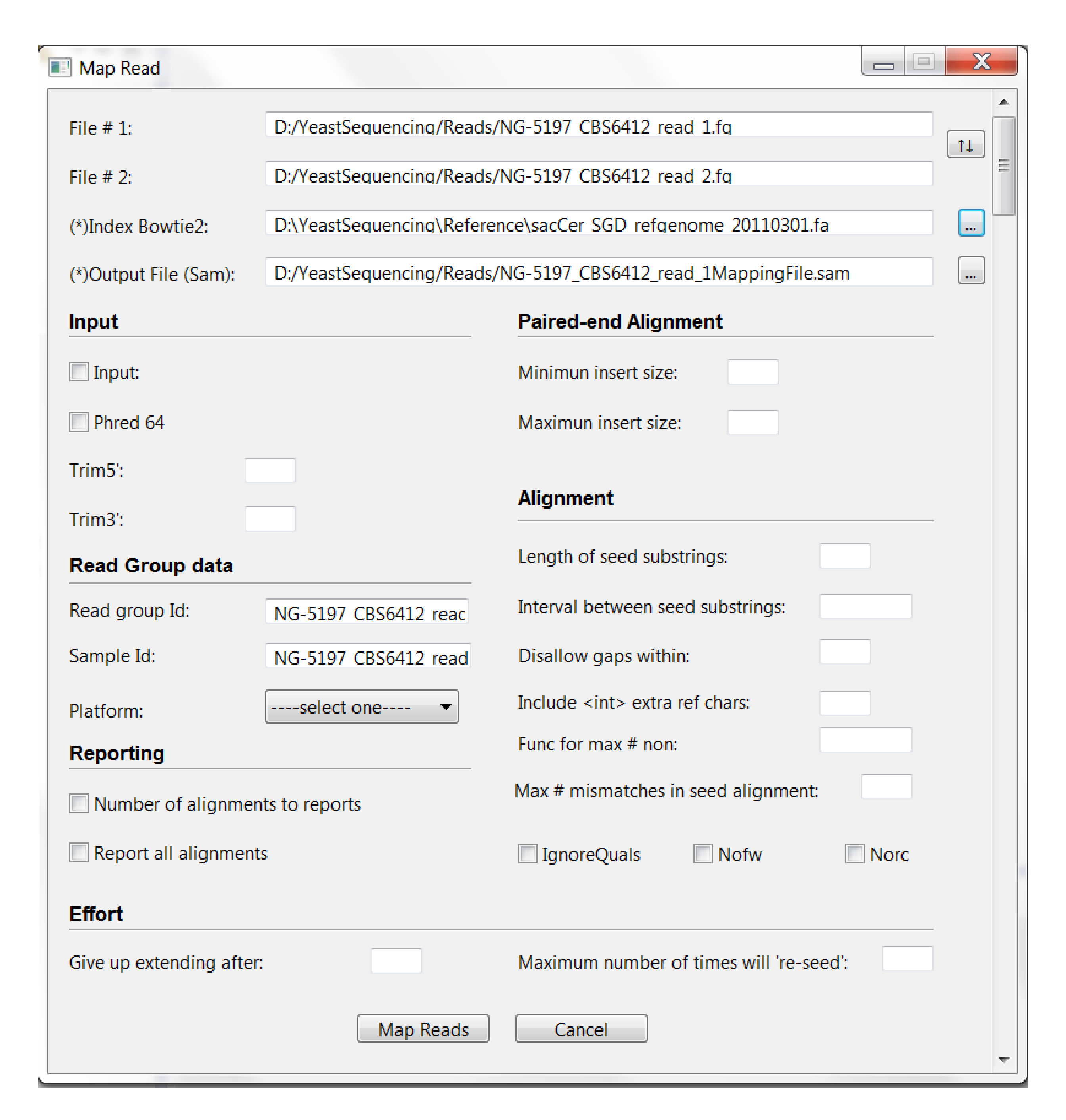
USING MAP READS

After completion of indexing the reference genome, you can execute the Map Reads function in NGSEP.

You will need your input files in FASTQ format (.fq or .fastq) uploaded in your eclipse IE. You can select a unique file or two files in case that you have complementary data. Doing a right click in the files, you can select the Map Reads function in the NGSEP menu.



Step one
Step two
Step Three



MAP READS PARAMETERS

1. **Input and Output Files**

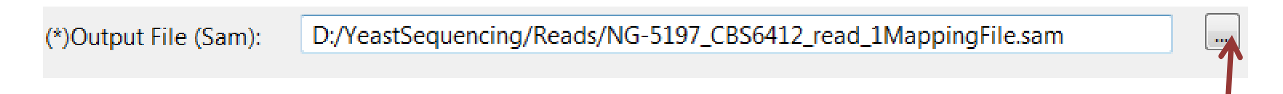
A. File #1 y File #2 fields show the path of your input files. You can also switch your input files using the browser option.

File # 1: D:/YeastSequencing/Reads/NG-5197_CBS6412_read_1.fq

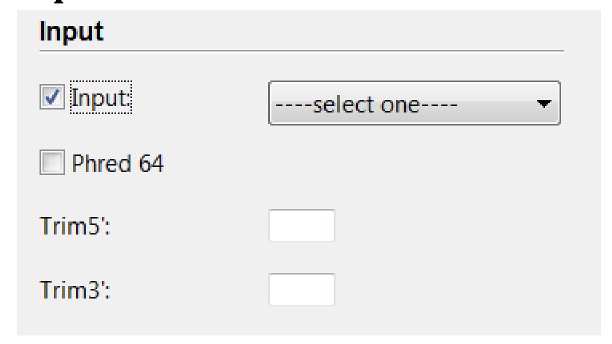
File # 2: D:/YeastSequencing/Reads/NG-5197_CBS6412_read_2.fq

B. (*)Index Bowtie2: Select the reference genome previously indexed by bowtie2 order. Next time that you open this screen you will see the last file that you entered.

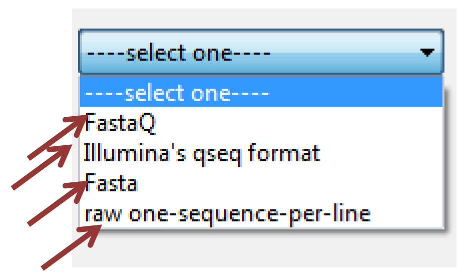
C. (*)Output File (Sam): Enter the name and the path where you want to save your output file.



2. Input



Select the Input flag if you know the format of your input files, and choose the adequate option.



The different file formats are explained in the chart below:

Input File Format	Flag	Description
FastaQ	**-q**	**Reads (specified with <m1>, <m2>, <s>) are FASTQ files. FASTQ files usually have extension .fq or .fastq. FASTQ is the default format. See also: --solexa-quals and --int-quals.**
Ilumina qseq format:	**--qseq**	**Reads (specified with <m1>, <m2>, <s>) are QSEQ files. QSEQ files usually end in _qseq.txt. See also: --solexa-quals and--int-quals.**
Fasta	**-f**	**Reads (specified with <m1>, <m2>, <s>) are FASTA files. FASTA files usually have extension .fa, .fasta, .mfa, .fna or similar. FASTA files do not have a way of specifying quality values, so when -f is set, the result is as if --ignore-quals is also set.**
Raw one-sequence-per-line:	**-r**	**Reads (specified with <m1>, <m2>, <s>) are files with one input sequence per line, without any other information (no read names, no qualities). When -r is set, the result is as if --ignore-quals is also set.**

3. Phred

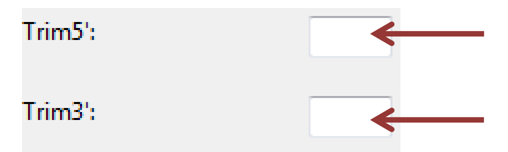
Phred 64

(--phred64)

Input qualities are ASCII chars equal to the Phred quality plus 64. This is also called the "Phred+64" encoding.

If you don't select the **--phred64** flag, the default will be **--phred33.** Input qualities are ASCII chars equal to the Phred quality plus 33. This is also called the "Phred+33" encoding, which is used by the very latest Illumina pipelines.

4. Trim



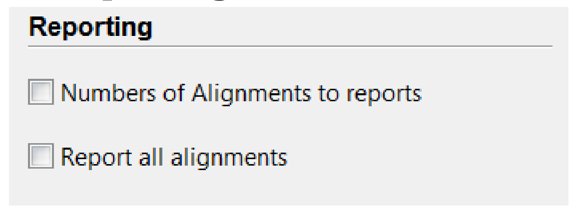
(---trim5)

Trim <int> bases from 5' (left) end of each read before alignment (default: 0).

(---trim3)

Trim <int> bases from 3' (right) end of each read before alignment (default: 0).

5. Reporting



The reporting mode allows for the search for one or more alignments and to report each one. Bowtie2 has three distinct reporting modes. The default mode is similar to the default reporting mode of many other read alignment tools, including BWA. It is also similar to Bowtie 1's -M alignment mode. In general, when we say that a read has an alignment, we mean that it has a valid alignment. When we say that a read has multiple alignments, we mean that it has multiple alignments that are valid and distinct from one another.

If you choose **Numbers of Alignments to report (-k mode),** you will find another field available where you can input the number of alignments you which to report.

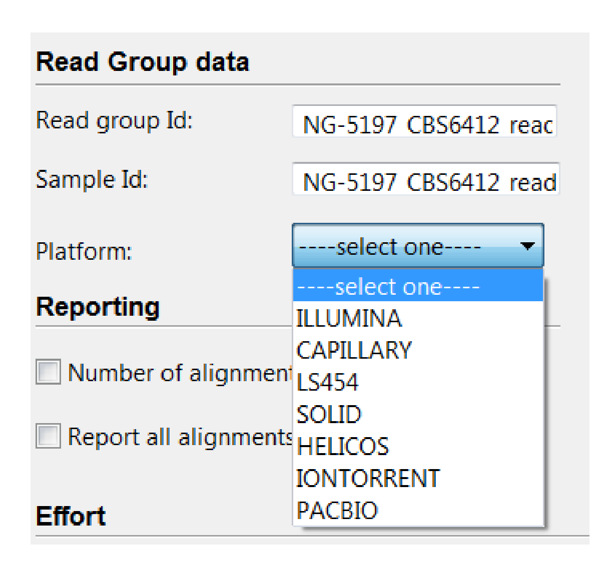


Using this option, Bowtie 2 searches for up to N distinct, valid alignments for each read, where N is the integer specified in the "Number of Alignments to report" field. If, for example, 2 is specified,

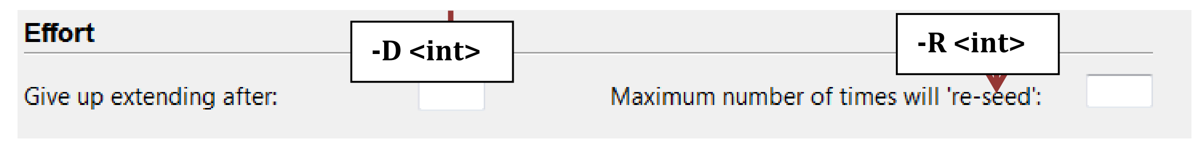
Bowtie 2 will search for at most 2 distinct alignments. It reports all alignments found, in descending order by alignment score. The alignment score for a paired-end alignment equals the sum of the alignment scores of the individual mates. Each reported read or pair alignment beyond the first has the SAM 'secondary' bit (which equals 256) set in its FLAGS field. See the SAM specification for details.

Bowtie 2 does not "find" alignments in any specific order, so for reads that have more than N distinct, valid alignments, Bowtie 2 does not guarantee that the N alignments reported are the best possible in terms of alignment score. Still, this mode can be effective and fast in situations where the user cares more about whether a read aligns (or aligns a certain number of times) than where exactly it originated.

6. Read Group data



7. Effort



-D <int>

By selecting the “Give up extending after” option (-D), you can choose the number of consecutive seed extension attempts that can "fail" before Bowtie 2 moves on, using the alignments found so far. A seed extension "fails" if it does not yield a new best or a new second-best alignment. The default value is 15.

-R <int>

With the “For reads w/ repetitive seeds” (-R) option you can choose the maximum number of times Bowtie 2 will "re-seed" reads with repetitive seeds. When "re-seeding," Bowtie 2 simply chooses a

new set of reads (same length, same number of mismatches allowed) at different offsets and searches for more alignments. A read is considered to have repetitive seeds if the total number of seed hits divided by the number of seeds that aligned at least once is greater than 300. In this case, the default value is 2.

8. Alignment

Alignment

Length of seed substrings:

Interval between seed substrings:

Disallow gaps within:

Include <int> extra ref chars:

Func for max # non:

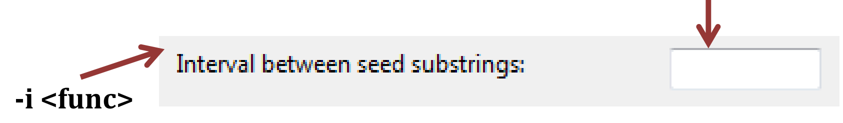
Max # mismatches in seed alignment:

IgnoreQuals Nofw Norc

Length of seed substrings:

-L <int>

Sets the length of the seed substrings to align during multiseed alignment. Smaller values make alignment slower but more sensitive. Default: the --sensitive preset is used by default, which sets -L to 20 both in --end-to-end mode and in --local mode.



-i <func>

Sets a function governing the interval between seed substrings to use during multiseed alignment. For instance, if the read has 30 characters, and seed length is 10, and the seed interval is 6, the seeds extracted will be:

Read: TAGCTACGCTCTACGCTATCATGCATAAAC

Seed 1 fw: TAGCTACGCT

Seed 1 rc: AGCGTAGCTA

Seed 2 fw: CGCTCTACGC

Seed 2 rc: GCGTAGAGCG

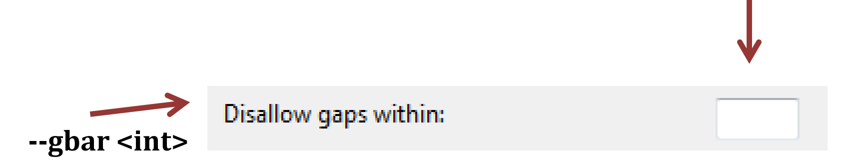
Seed 3 fw: ACGCTATCAT

Seed 3 rc: ATGATAGCGT

Seed 4 fw: TCATGCATAA

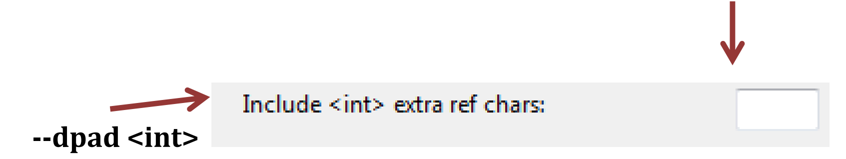
Seed 4 rc: TTATGCATGA

Since it's best to use longer intervals for longer reads, this parameter sets the interval as a function of the read length, rather than a single one-size-fits-all number. For instance, specifying -i S, 1, 2.5 sets the interval function f to f(x) = 1 + 2.5 * sqrt(x), where x is the read length. See also: setting function options. If the function returns a result less than 1, it is rounded up to 1. Default: the --sensitive preset is used by default, which sets -i to S,1,1.15 in --end-to-end mode to -i S,1,0.75 in --local mode.



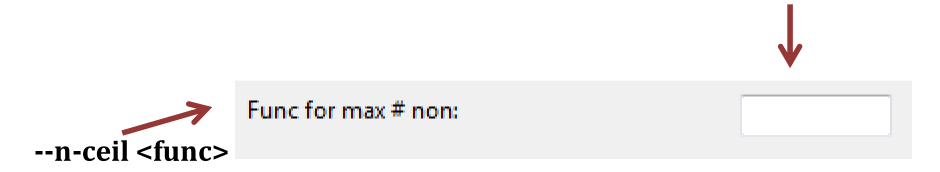
--gbar <int>

Disallow gaps within <int> positions of the beginning or end of the read. Default: 4.



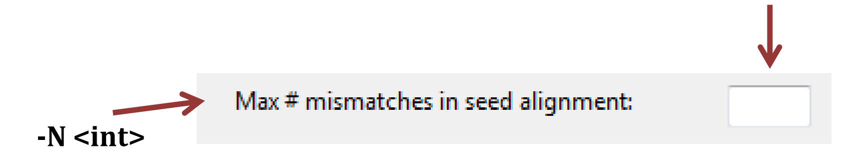
--dpad <int>

"Pads" dynamic programming problems by <int> columns on either side to allow gaps. Default: 15



--n-ceil <func>

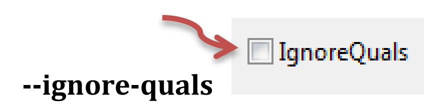
Sets a function governing the maximum number of ambiguous characters (usually Ns and/or .s) allowed in a read as a function of read length. For instance, specifying -L, 0, 0.15 sets the N-ceiling function f to f(x) = 0 + 0.15 * x, where x is the read length. See also: setting function options. Reads exceeding this ceiling are filtered out. Default: L, 0, 0.15.



-N <int>

Sets the number of mismatches to allowed in a seed alignment during multiseed alignment. Can be set to 0 or 1. Setting this higher makes alignment slower (often much slower) but increases sensitivity. Default: 0.

.



--ignore-quals

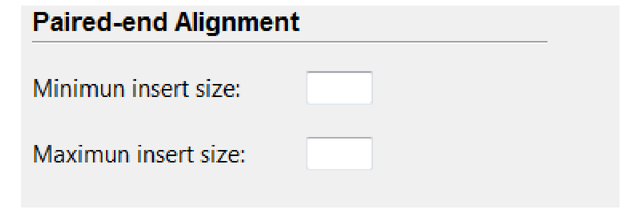
When calculating a mismatch penalty, always consider the quality value at the mismatched position to be the highest possible, regardless of the actual value. I.e. input is treated as though all quality values are high. This is also the default behavior when the input doesn't specify quality values (e.g. in -f, -r, or -c modes).



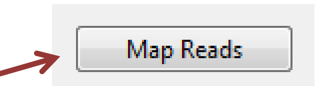
--nofw/--norc

If --nofw is specified, bowtie2 will not attempt to align unpaired reads to the forward (Watson) reference strand. If --norc is specified, bowtie2 will not attempt to align unpaired reads against the reverse-complement (Crick) reference strand. In paired-end mode, --nofw and --norc pertain to the fragments; i.e. specifying --nofw causes bowtie2 to explore only those paired-end configurations corresponding to fragments from the reverse-complement (Crick) strand. Default: both strands enabled.

9. Paired-end Alignment



Finally, the **Map Reads** is the one that performs the process invoking Bowtie2, after validating the data entered.



Final Results for Map Read:

-At the end of the process you will generate a .Bam file that you named previously with all the reads matched against the reference.

Sort Alignment

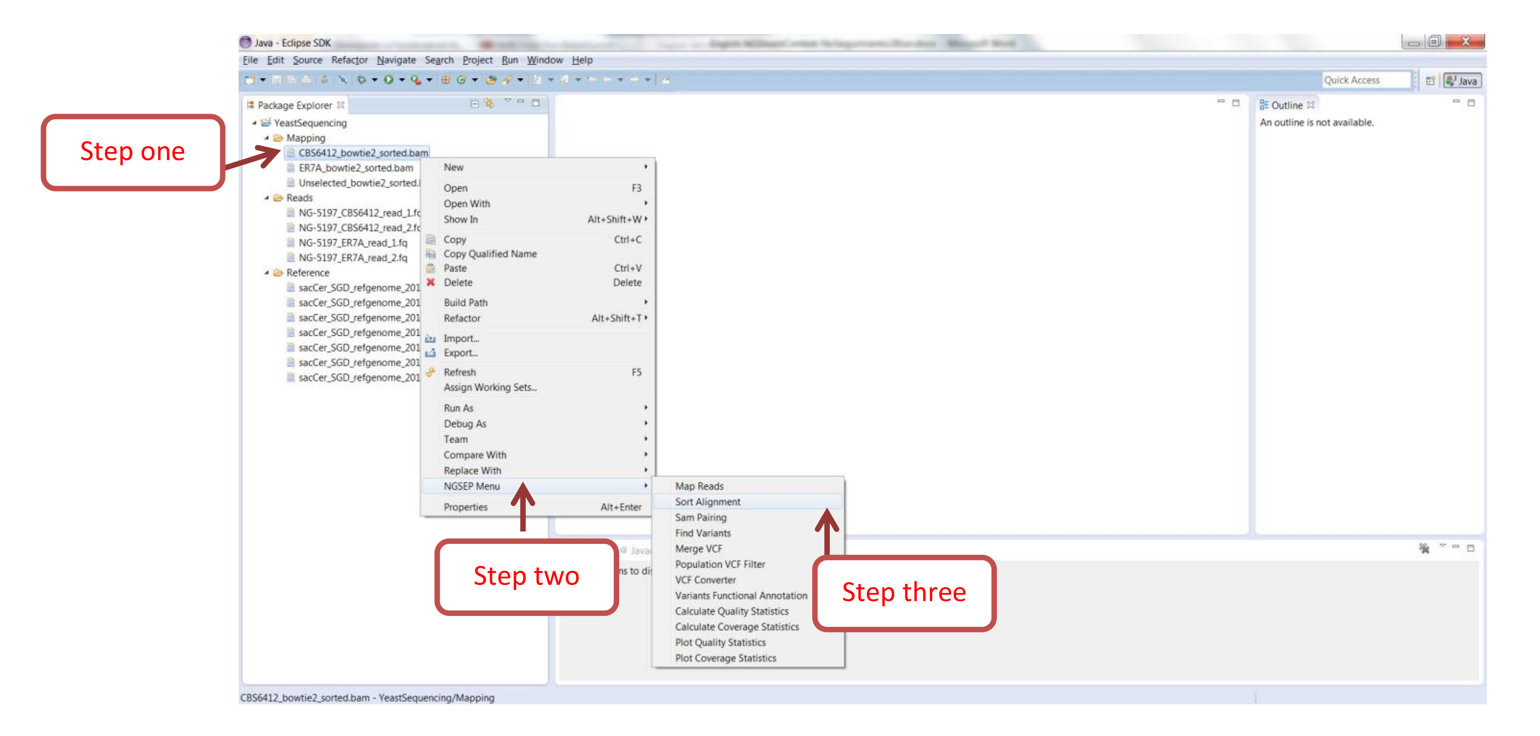
This option sorts the Bam file, which is the output of the **Map reads** function. This process is required because sequencers such as Illumina, 454 and Sanger among others, produce files that match randomly in the genome. Sort Alignment uses internally Picards Tools, a library which already contains an option for this purpose.

INPUT FILES

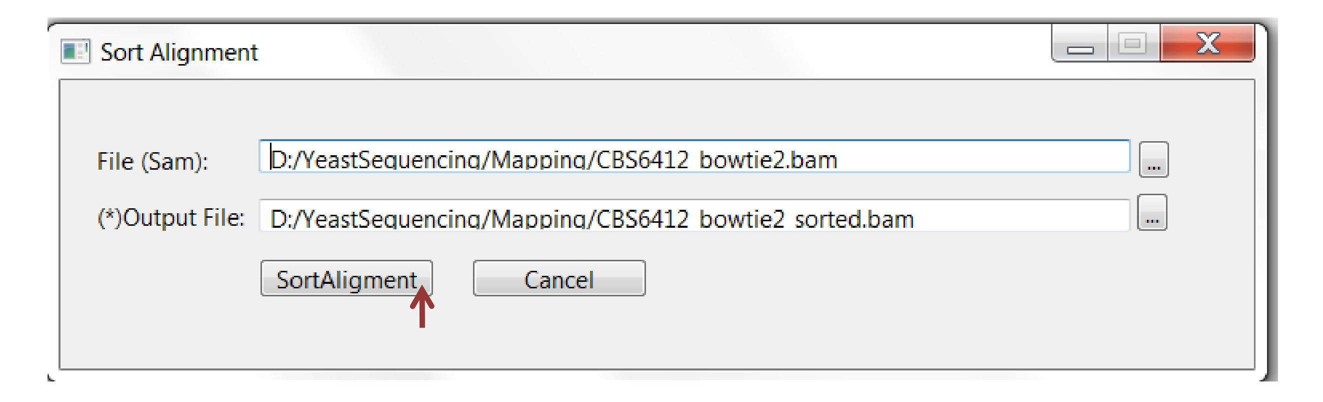
.Bam: Text format tab delimited file which consist in a header section, that is optional and a section of alignment. The header begins with @ while the alignment lines don't. Each aligned line has 11 optional information fields that make it flexible.

ACCESS TO SORT ALIGNMENT

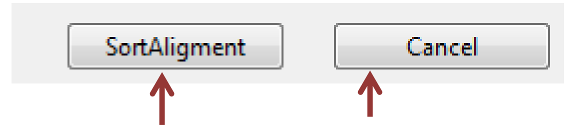
1. The first step in order to access to Sort Alignment after installing Eclipse and NGSEP is having the Bam file.
2. Click on the .Bam file, and choose the **Sort Alignment** option from the NGSEP menu.
3. Make sure that you only select one Bam file.



4. The first field is the **File (Sam),** which shows the path of the input file that you browsed. The next one is **(*) Output file:** This text field holds the same input name with the addition of the word sorted just before the extension. Also, you can change the output destiny directory. Our advice is to use the same directory because further processes will require them.



5. Use the button with the label Sort Alignment to execute if you want to close the window click on cancel.



Final Result for Sort Alignment:

-At the end of the process you will see a similar file than the input, but organized and ready to continue with the pipeline.

Variants Detector

This is the main functionality of NGSEP, which using a Bam file against a reference genome will detect different genomic variations such as: SNPs, CNVs and structural variants.

INPUT FILES

- **BAM:** Text format tab delimited which consist in a header section, that is optional and a section of alignment. The header begins with @ while the alignment lines don't. Each aligned line has 11 optional information fields that make it flexible. For more information see: http://samtools.sourceforge.net/

OUTPUT FILES

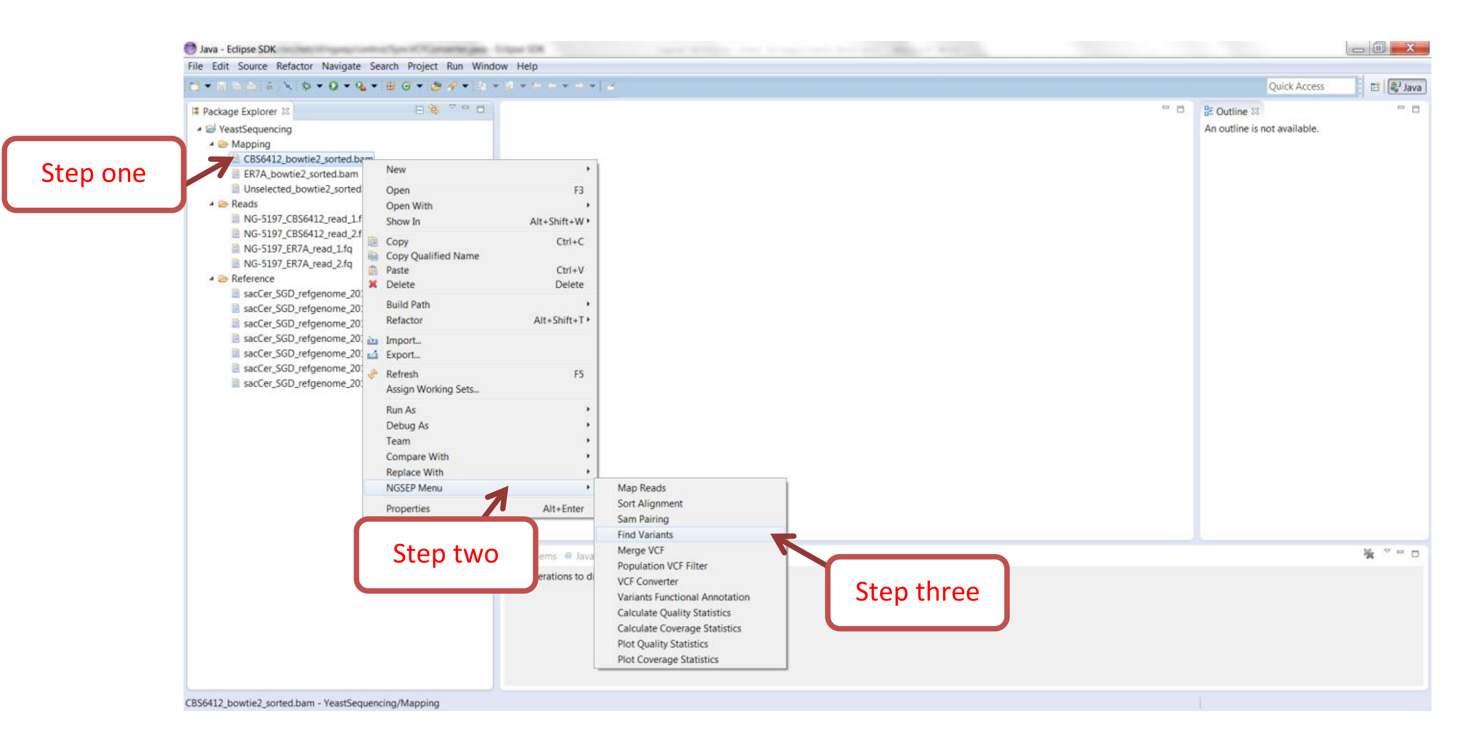
- **VCF** (Variant call format): Is flexible and extensible file for variation data such as (SNPs), small INDELs, CNVs and structural variants. For more information of the file format see: http://www.1000genomes.org/wiki/Analysis/Variant%20Call%20Format/vcf-variant-call-format-version-41

- **Gff:** General format of characteristics created by Sanger, composed by 9 mandatory fields separated by tabs. For more information see: http://www.sanger.ac.uk/resources/software/gff/spec.html

- **CNVs format**: For copy number variations

ACCESS TO VARIANTS DETECTOR

1. The first step in order to access to Variants Detector after installing Eclipse and NGSEP is having the Sorted Bam file.
2. Click on the sorted Bam file, and choose the **Find Variants option** from the NGSEP menu.
3. Make sure that the selected file is a Sorted Bam File otherwise the process will not work.



Screen Variants Detector

Variants Detector

(*)File : D:/YeastSequencing/Mapping/CBS6412_bowtie2_sorted.bam

(*)Reference File: D:\YeastSequencing\Reference\sacCer_SGD_refgenome_20110301.fa

(*)Output File Prefix: D:/YeastSequencing/Mapping/CBS6412_bowtie2

Find Variants

Execution Parameters

Skip Repetitive Regions Detection

Skip New CNV Detection

Skip Structural Variants Detection

Skip SNVs Detection

CNVs Detection Parameters

Genome Size:

Bin Size: 100

SNVs Detection Parameters

Genomic Location:

Heterozygosity Rate: 0.0010

Minimun Genotype Quality Score: 40

Maximun Base Quality Score: 30

Alternative Allele Coverage: Min: Max:

Ignore Lower Case Reference

Include Secondary Alignments

Maximun Alignment Per Start Position: 2

Ignore Bases 5': 0

Ignore Bases 3': 0

Common Parameters

Ploidy: 2

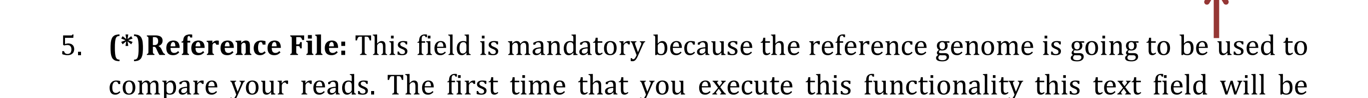
(*)Sample Id: CBS6412

Known CNVs File

Known Variants File

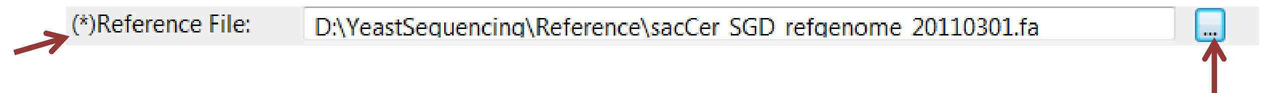
Find Variants Cancel

4. **(*)File:** In this field you can see the path of the sorted Bam file that you selected (It could be the output file of the Sort Alignment of NGSEP). Note that you can also use the browser on the right in case you want to change the input file.



5. **(*)Reference File:** This field is mandatory because the reference genome is going to be used to compare your reads. The first time that you execute this functionality this text field will be

blank, you must browse for a fasta file with the reference genome. However, for further executions the field will display the last reference used.



6. **(*)Output File Prefix:** This field refers to the output files you will generate with this function. In total you will generate 3 files that will be named with the same output prefix that you type. The generated files will be: **VFC:** For SNPs and Small indels, **CNV:** For Copy Number Variations. **GFF**: For SNVs and large indels. You can change the prefix and the destination directoryof your output file, using the browser on the right.

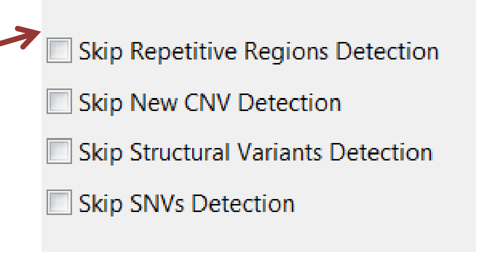
(*)Output File Prefix: C:/Users/jcquintero/workspace/TestingNGSEP/X4035 5000 bfast d5

Notice **that the output directory suggested is the same of the input file as well as the name of the tested sample.**

(*)Output File Prefix: D:/YeastSequencing/Mapping/CBS6412 bowtie2

7. Execution Parameters

This section is compose by 4 parameters that represent the whole variant detection process.



-Skip Repetitive Regions Detection: this option is intended to set aside repetitive regions for detecting the other genomic variants.

-Skip New CNV Detection: This option is intended to set aside CNVs for detecting the other genomic variants.

-Skip SNVs Detection: This option is intended to set aside SNVs or SNPs for detecting the other genomic variants.

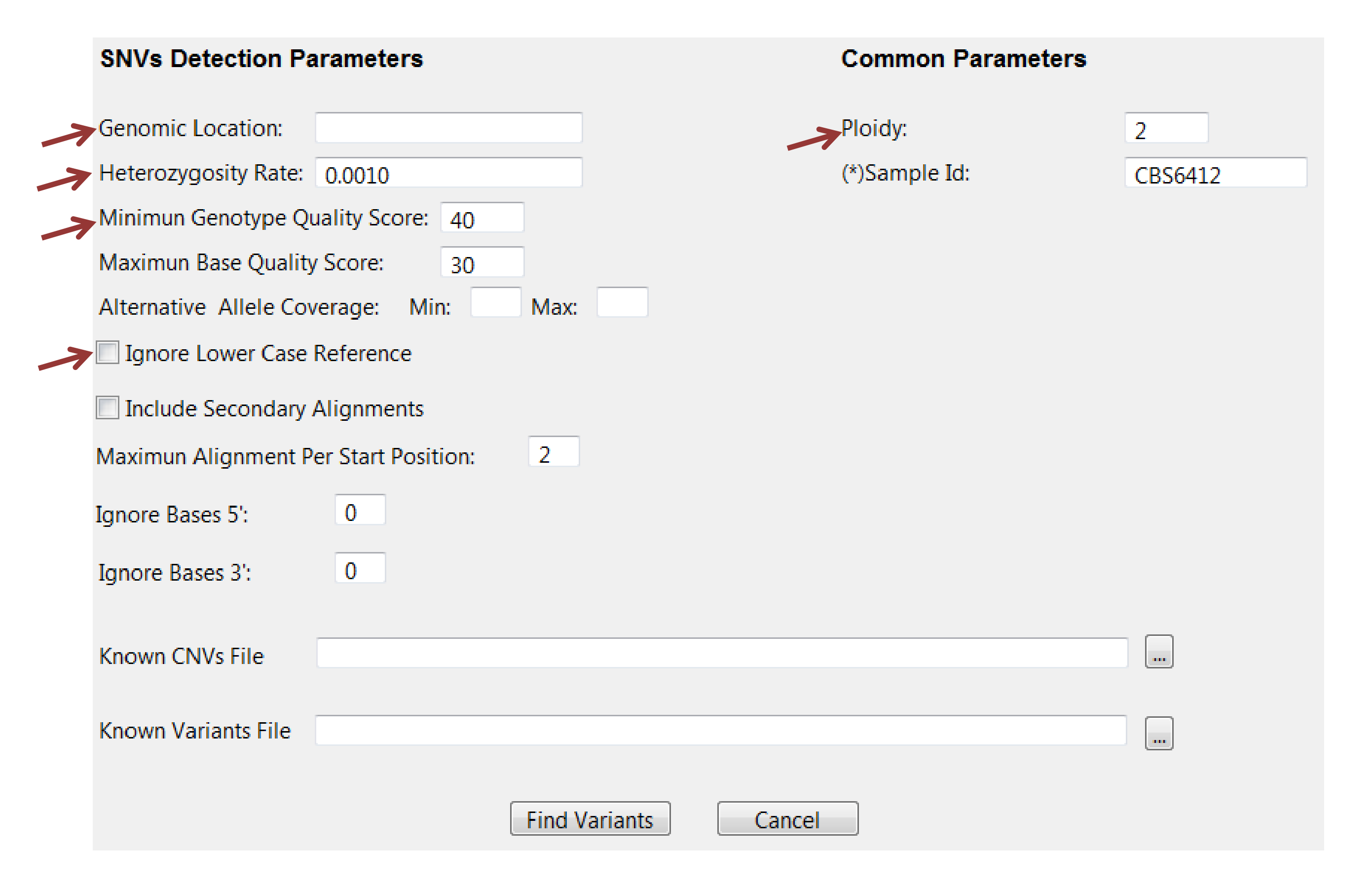
-Skip Structural Variants Detection: This option is intended to set aside structural variants for detecting genomic variants such us: Insertions, deletions, inversions.

Note: If you don't select any option from **Execution Parameters** NGSEP will execute all the findings of **variants detector.**

8. SNVs Detection Parameters

This section is composed by parameters that represent many adjustments that can improve the SNVs detection.

Note: By default some fields can hold values, however if you are aware about their meaning you can change on demand according to your sample on research.



-Genomic Location (optional): In this field, enter a specific location in the genome in order to detect SNPs. This is the format accepted: **'chr21:33,031,197-33,041,570'.**

Note: you must be aware of the number of chromosomes and range of detection.

-Heterozygosity Rate: This field is intended to enter the probability of finding in every certain position an heterozygous SNPs.

-Minimun Genotype Quality Score: Indicate the minimum accepted value of probability to consider an error (Phred Score).

-Maximun Base Quality Score: Maximum score allowed by allele.

-Alternative Allele Coverage: Maximum and minimum number of alleles that can present a position.

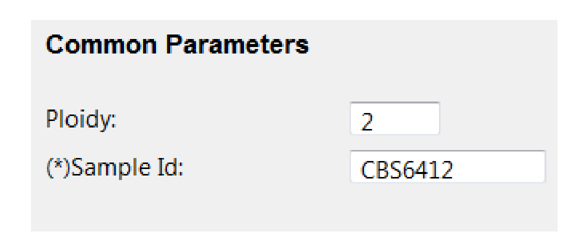
-**Ignore Lower Case References:** Select this option if you want to skip bases in lower case.

-**Maximun Alignment Per Start Position:** Use this quality filter to allows to correct some errors produced by PCR Amplificaction Artefacts.

-**Known CNVs File**: In this file you can enter the path of a CNV file that can be used in you detection.

9. Common Parameters

This section holds some parameters that are related to all processes in the Variants detection.



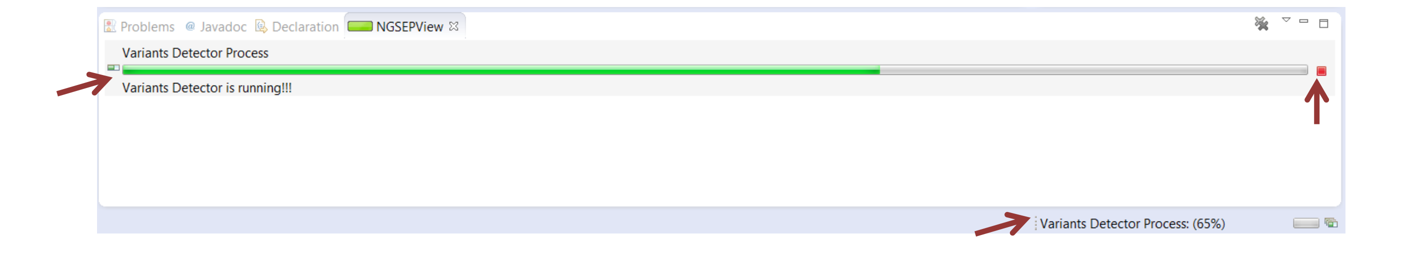
-Ploidy: For Haploid type 1, for diploid type 2.

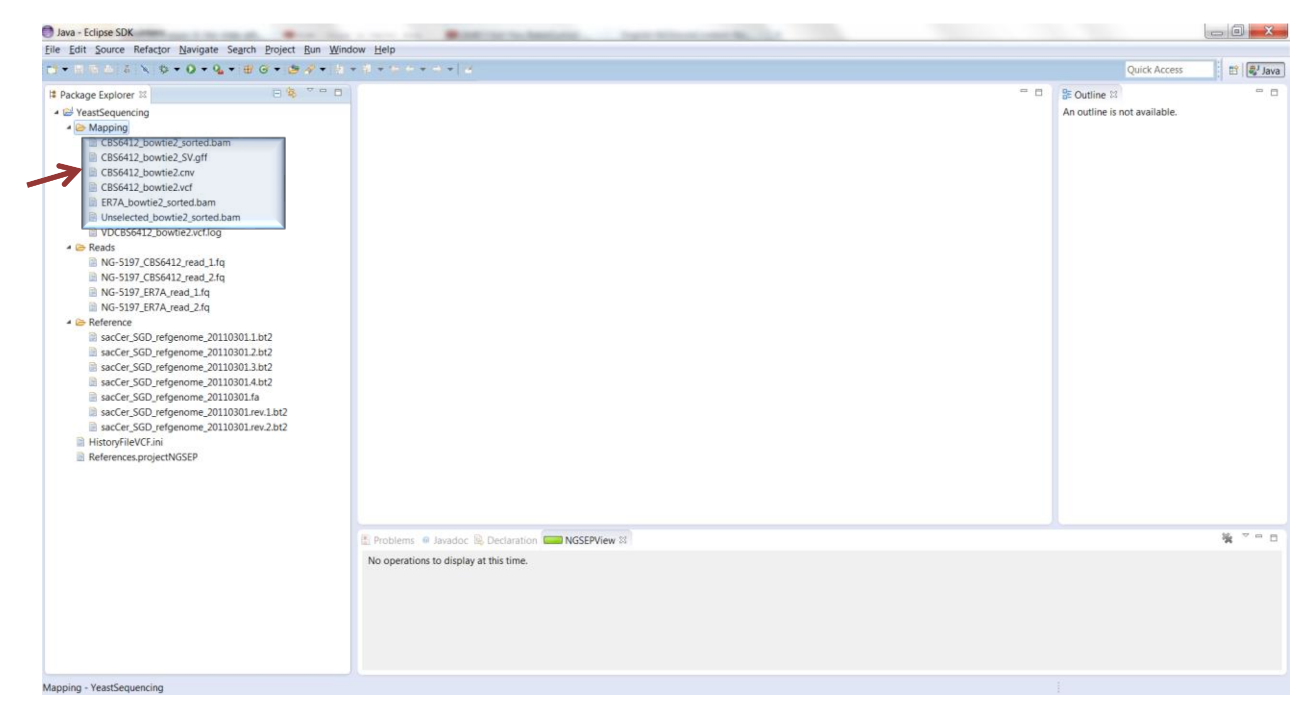
-Sample ID: You can type a specific ID to label the header of the **VFC**.

10. 

Use the button with the label (**Find Variants**) to execute if you want to close the window click on cancel.

Note: When you execute the variants detector, a progress bar will be displayed on the bottom, it represents the percentage of completed process. This is important because many times this process can takes several minutes depending on how complex is your organism. If you want to stop the process you are able to do it by pressing the red button in the right side of the progress view. At the end of the process you will see the output files in the directory that you selected.



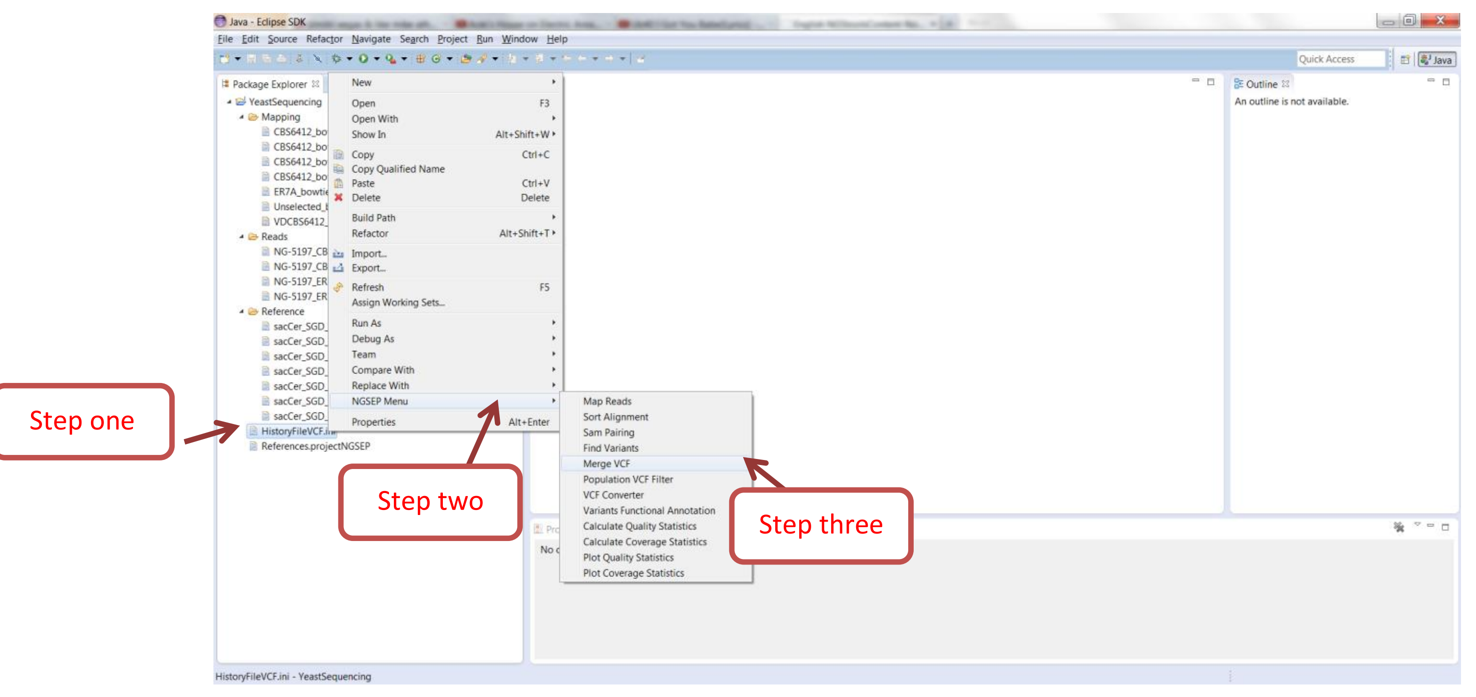


Merge VCF

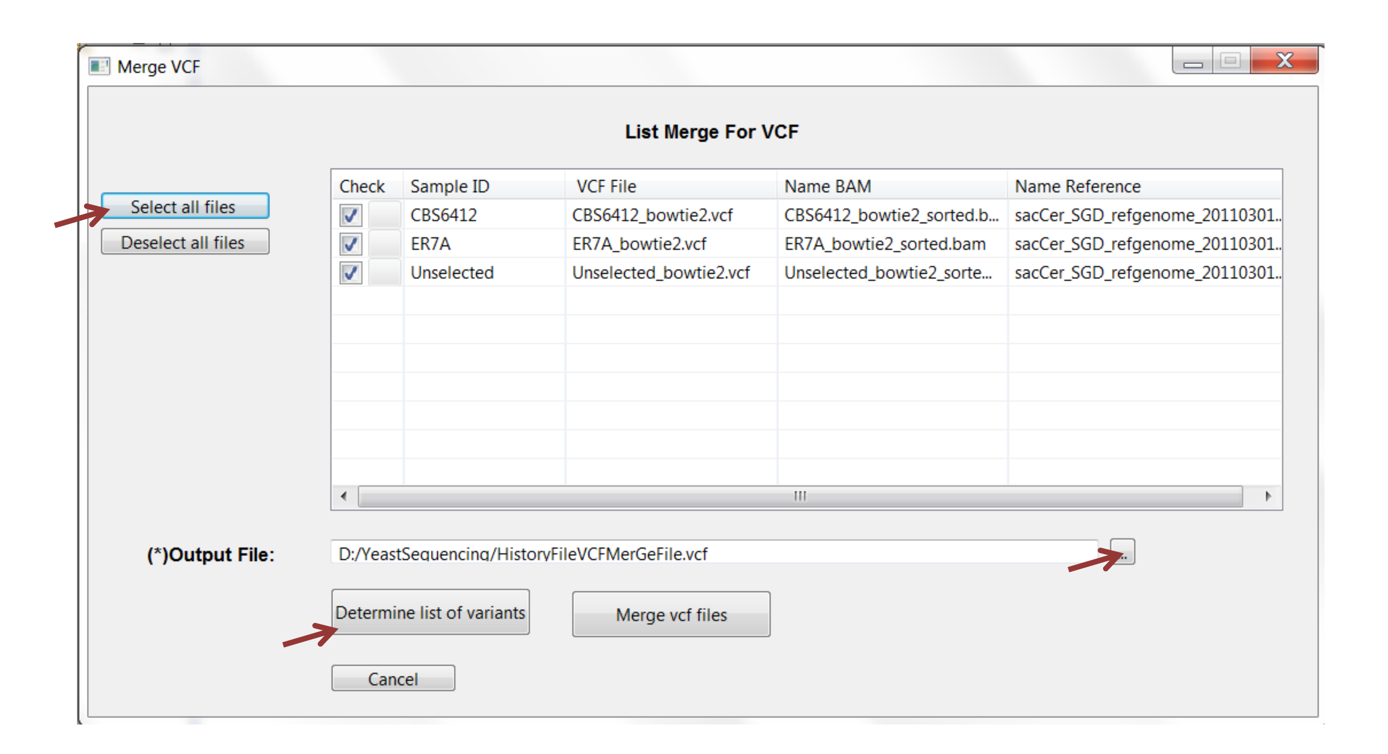
This process is divided in two phases; the first one is intended to determine a list of variants found in at least one of the VCF files that were generated in the variants detection process, creating one common VCF file. Afterwards the process requires running again Variants Detector for every sample but using the mentioned common file in the known variants field. Finally you will be able to merge those new VCF files, showing the inheritance from parents to offspring in terms of changing alleles.

ACCESS TO MERGE VCF

1. The first step is making sure that you have the detector variants history file with three samples, otherwise you have to execute Variants detector for the target samples.
2. Click on the file named HistoryFileVCF.ini, and choose the **Merge VCF** option from the NGSEP menu.
3. Make sure that the selected file is Detector Variants history with more than three samples otherwise the process will not work properly.



FIRST OPTION WITH BUTTON DETERMINE LIST OF VARIANTS



(*) Output File : this field means the output file Merge VCF and is mandatory.
Select all files: This option is equivalent to select all rows.
Deselect all files: This option is equivalent to deselect all rows.
Determine list of variants: This option is used to mix in one VCF without genotypes, all variant alleles found in at least one of the files.
Merge Vcf files: This option is used to mix in a single VCF file all genomic variants found in VCF file, matching each corresponding mutation with their genotype.

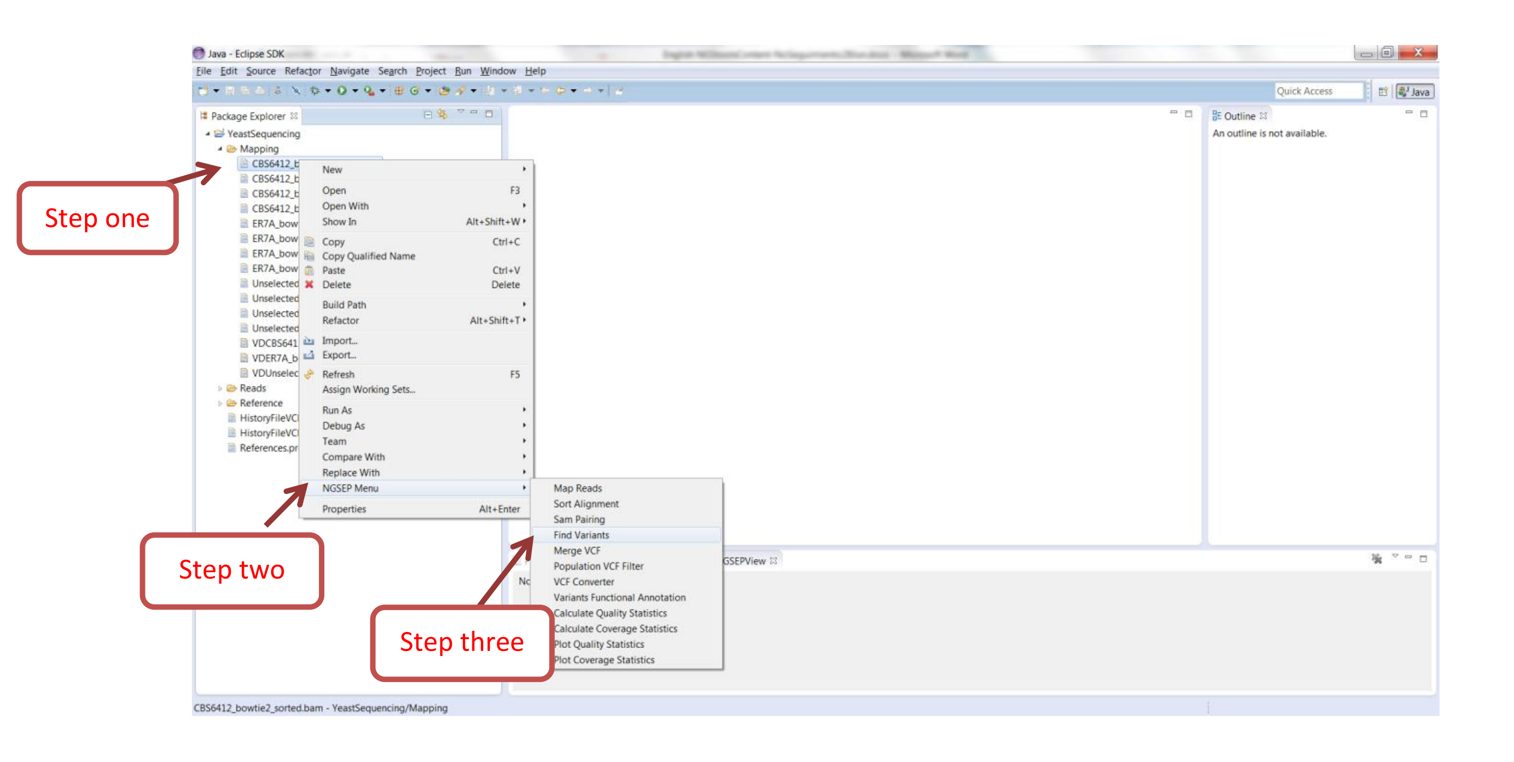
Output file using the "Determine list of variants" option

```
D:\YeastSequencing\HistoryFileVCFMerGeFile.vcf - Notepad++
##fileformat=VCFv4.1
##INFO=<ID=CNV,Number=1,Type=Integer,Description="Number of samples with CNVs around this variant">
##INFO=<ID=TA,Number=1,Type=String,Description="Variant annotation based on a gene model">
##INFO=<ID=TID,Number=1,Type=String,Description="Id of the transcript related to the variant annotation">
##INFO=<ID=TGN,Number=1,Type=String,Description="Name of the gene related to the variant annotation">
##INFO=<ID=TCO,Number=1,Type=Float,Description="One based codon position of the start of the variant. The decimal is the codon position">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GL,Number=G,Type=Float,Description="Genotype likelihoods">
##FORMAT=<ID=GP,Number=G,Type=Integer,Description="Genotype posterior probabilities">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype quality">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read depth">
##FORMAT=<ID=AC,Number=A,Type=Integer,Description="Counts for observed alleles">
##FORMAT=<ID=AAC,Number=.,Type=Integer,Description="Counts for all possible alelles">
#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO	FORMAT	None
chrI	90	.	A	C	71	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	100	.	GG	GA,GAT	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	103	.	C	T	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	107	.	C	A	192	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	114	.	T	A	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	115	.	C	A	55	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	117	.	A	G	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	118	.	C	T	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	136	.	G	A	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	138	.	CT	CCAC,CTT	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	141	.	C	T	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	172	.	A	G	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	176	.	CGA	CA,CCA	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	181	.	C	T	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	241	.	C	T	71	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	249	.	T	C	47	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	250	.	G	A	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	254	.	CTT	CT	98	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	257	.	A	C	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	262	.	A	G	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	266	.	T	A	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	268	.	A	C	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	269	.	C	A	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
chrI	270	.	C	T	255	.	.	GT:GL:GP:GQ:DP:AC	./.:.:.:0:0:.
```

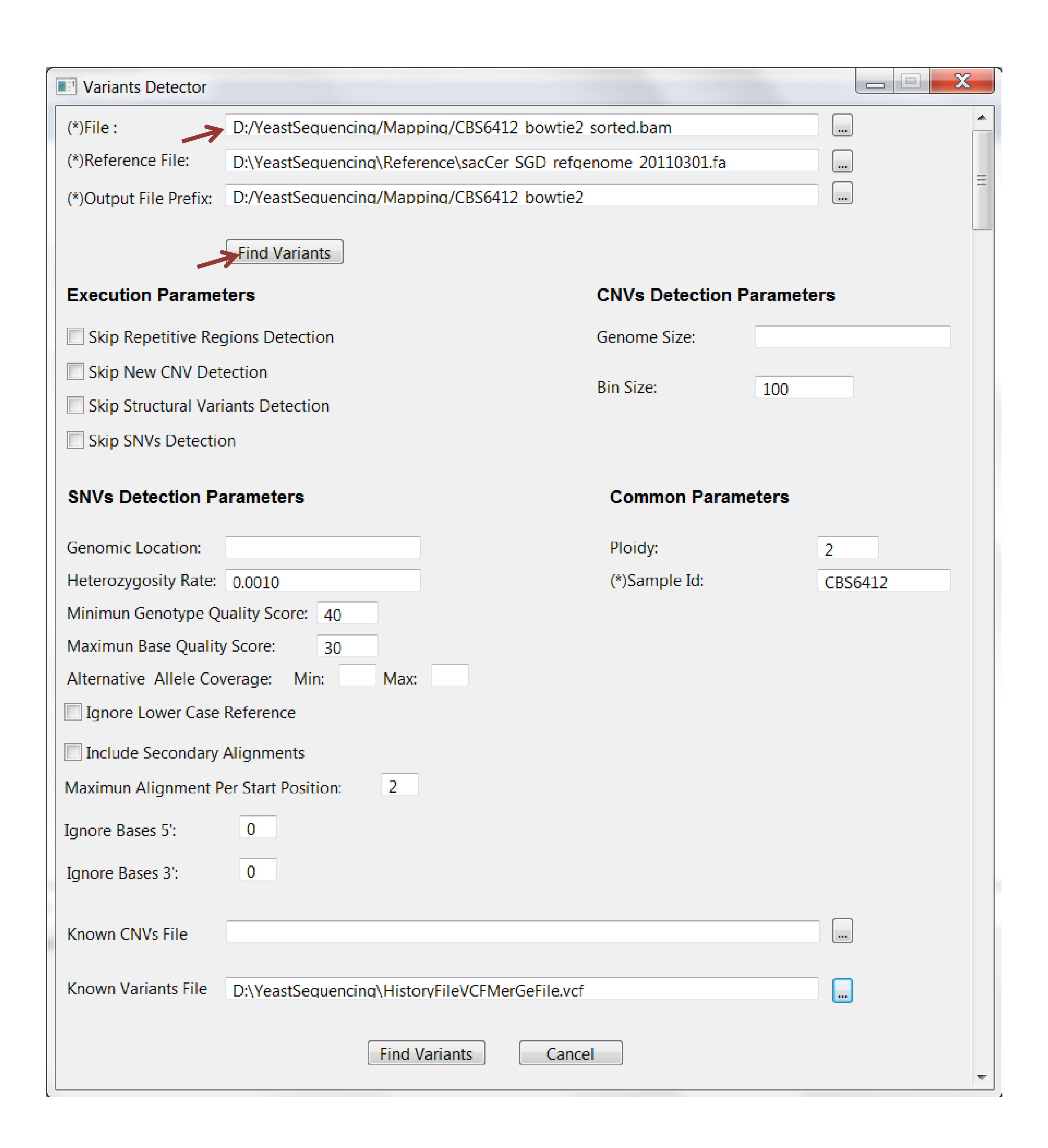
This option is used to generate VCF file information for genetic mutations that are present at least in one of the files selected by the user.

The generated file is important for the execution of the second process (Merge VCF), because this file is necessary to execute again variants detector for each sample in order to associate the variant allele with their corresponding sample genotype.

After finishing the list variants process, proceed to right click on each BAM file using the known variants field to run variants detector as follows:

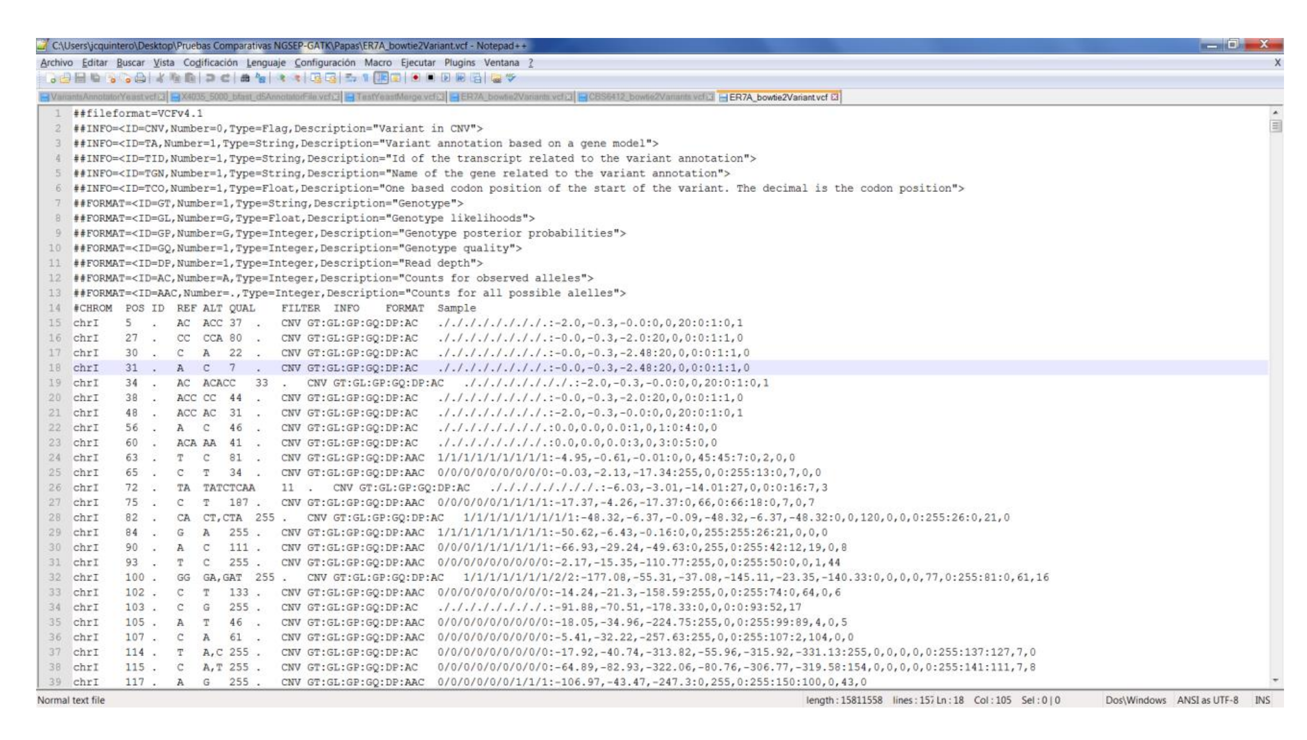


Step one
Step two
Step three



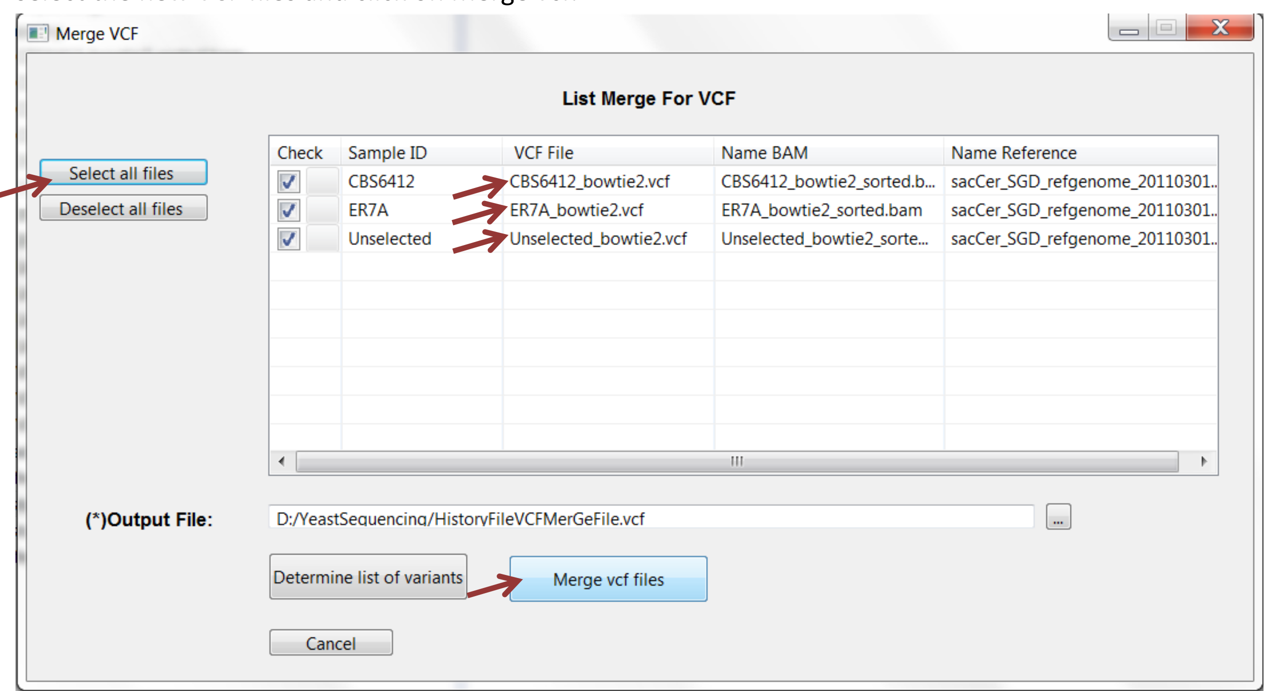
Variants Detector
(*)File :
D:/YeastSequencing/Mapping/CBS6412_bowtie2_sorted.bam
(*)Reference File:
D:\YeastSequencing\Reference\sacCer_SGD_refgenome_20110301.fa
(*)Output File Prefix:
D:/YeastSequencing/Mapping/CBS6412_bowtie2
Find Variants
Execution Parameters
Skip Repetitive Regions Detection
Skip New CNV Detection
Skip Structural Variants Detection
Skip SNVs Detection
CNVs Detection Parameters
Genome Size:
Bin Size:
100
SNVs Detection Parameters
Genomic Location:
Heterozygosity Rate:
0.0010
Minimun Genotype Quality Score:
40
Maximun Base Quality Score:
30
Alternative Allele Coverage:
Min:
Max:
Ignore Lower Case Reference
Include Secondary Alignments
Maximun Alignment Per Start Position:
2
Ignore Bases 5':
0
Ignore Bases 3':
0
Common Parameters
Ploidy:
2
(*)Sample Id:
CBS6412
Known CNVs File
Known Variants File
D:\YeastSequencing\HistoryFileVCFMerGeFile.vcf
Find Variants
Cancel

Generated file

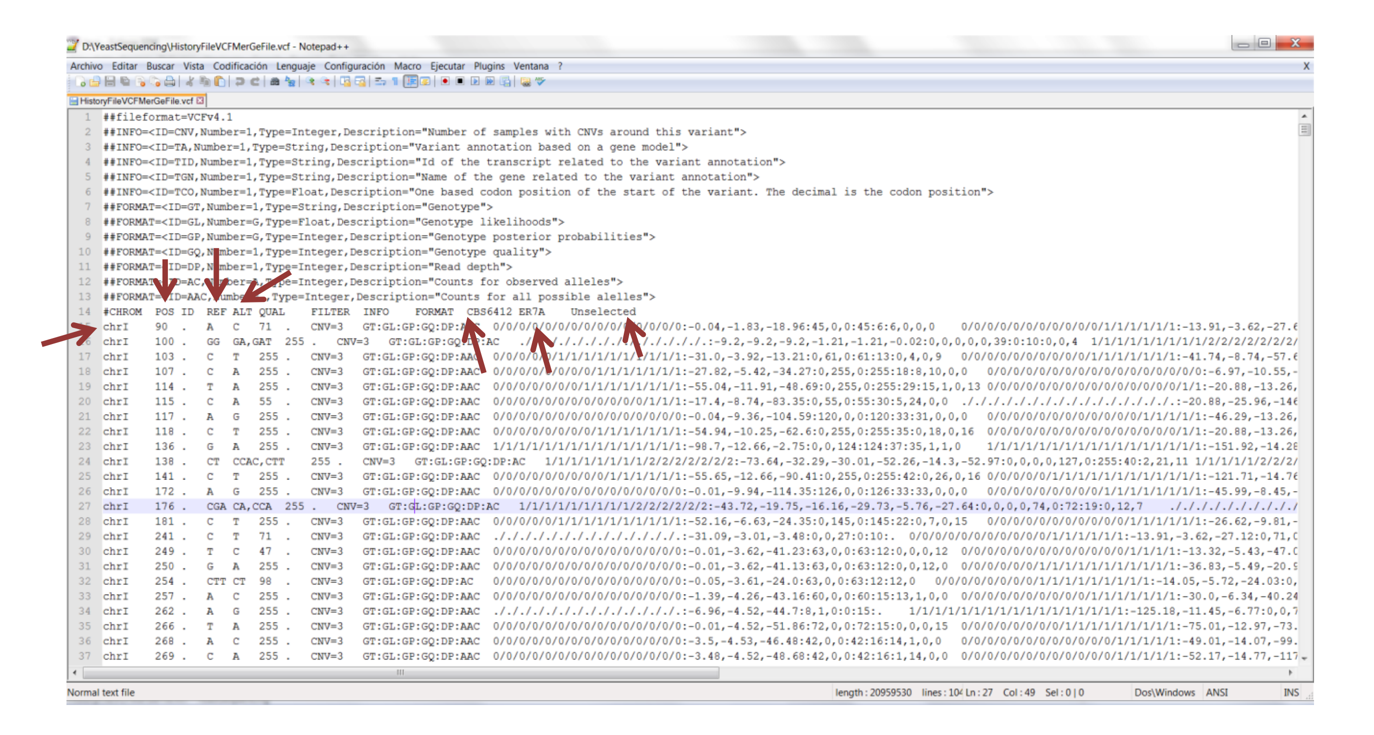


```
##fileformat=VCFv4.1
##INFO=<ID=CNV,Number=0,Type=Flag,Description="Variant in CNV">
##INFO=<ID=TA,Number=1,Type=String,Description="Variant annotation based on a gene model">
##INFO=<ID=TID,Number=1,Type=String,Description="Id of the transcript related to the variant annotation">
##INFO=<ID=TGN,Number=1,Type=String,Description="Name of the gene related to the variant annotation">
##INFO=<ID=TCO,Number=1,Type=Float,Description="One based codon position of the start of the variant. The decimal is the codon position">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GL,Number=G,Type=Float,Description="Genotype likelihoods">
##FORMAT=<ID=GP,Number=G,Type=Integer,Description="Genotype posterior probabilities">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype quality">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read depth">
##FORMAT=<ID=AC,Number=A,Type=Integer,Description="Counts for observed alleles">
##FORMAT=<ID=AAC,Number=.,Type=Integer,Description="Counts for all possible alelles">
#CHROM  POS ID  REF ALT QUAL    FILTER  INFO    FORMAT  Sample
chrI    5   .   AC  ACC 37  .   CNV GT:GL:GP:GQ:DP:AC   ./././././././././.:-2.0,-0.3,-0.0:0,0,20:0:1:0,1
chrI    27  .   CC  CCA 80  .   CNV GT:GL:GP:GQ:DP:AC   ./././././././././.:-0.0,-0.3,-2.0:20,0,0:0:1:1,0
chrI    30  .   C   A   22  .   CNV GT:GL:GP:GQ:DP:AC   ./././././././././.:-0.0,-0.3,-2.48:20,0,0:0:1:1,0
chrI    31  .   A   C   7   .   CNV GT:GL:GP:GQ:DP:AC   ./././././././././.:-0.0,-0.3,-2.48:20,0,0:0:1:1,0
chrI    34  .   AC  ACACC   33  .   CNV GT:GL:GP:GQ:DP:AC   ./././././././././.:-2.0,-0.3,-0.0:0,0,20:0:1:0,1
chrI    38  .   ACC CC  44  .   CNV GT:GL:GP:GQ:DP:AC   ./././././././././.:-0.0,-0.3,-2.0:20,0,0:0:1:1,0
chrI    48  .   ACC AC  31  .   CNV GT:GL:GP:GQ:DP:AC   ./././././././././.:-2.0,-0.3,-0.0:0,0,20:0:1:0,1
chrI    56  .   A   C   46  .   CNV GT:GL:GP:GQ:DP:AC   ./././././././././.:0.0,0.0,0.0:1,0,1:0:4:0,0
chrI    60  .   ACA AA  41  .   CNV GT:GL:GP:GQ:DP:AC   ./././././././././.:0.0,0.0,0.0:3,0,3:0:5:0,0
chrI    63  .   T   C   81  .   CNV GT:GL:GP:GQ:DP:AAC  1/1/1/1/1/1/1/1/1:-4.95,-0.61,-0.01:0,0,45:45:7:0,2,0,0
chrI    65  .   C   T   34  .   CNV GT:GL:GP:GQ:DP:AAC  0/0/0/0/0/0/0/0/0:-0.03,-2.13,-17.34:255,0,0:255:13:0,7,0,0
chrI    72  .   TA  TATCTCAA    11  .   CNV GT:GL:GP:GQ:DP:AC   ./././././././././.:-6.03,-3.01,-14.01:27,0,0:0:16:7,3
chrI    75  .   C   T   187 .   CNV GT:GL:GP:GQ:DP:AAC  0/0/0/0/0/1/1/1/1:-17.37,-4.26,-17.37:0,66,0:66:18:0,7,0,7
chrI    82  .   CA  CT,CTA  255 .   CNV GT:GL:GP:GQ:DP:AC   1/1/1/1/1/1/1/1/1:-48.32,-6.37,-0.09,-48.32,-6.37,-48.32:0,0,120,0,0,0:255:26:0,21,0
chrI    84  .   G   A   255 .   CNV GT:GL:GP:GQ:DP:AAC  1/1/1/1/1/1/1/1/1:-50.62,-6.43,-0.16:0,0,255:255:26:21,0,0,0
chrI    90  .   A   C   111 .   CNV GT:GL:GP:GQ:DP:AAC  0/0/0/1/1/1/1/1/1:-66.93,-29.24,-49.63:0,255,0:255:42:12,19,0,8
chrI    93  .   T   C   255 .   CNV GT:GL:GP:GQ:DP:AAC  0/0/0/0/0/0/0/0/0:-2.17,-15.35,-110.77:255,0,0:255:50:0,0,1,44
chrI    100 .   GG  GA,GAT  255 .   CNV GT:GL:GP:GQ:DP:AC   1/1/1/1/1/1/1/2/2:-177.08,-55.31,-37.08,-145.11,-23.35,-140.33:0,0,0,0,77,0:255:81:0,61,16
chrI    102 .   C   T   133 .   CNV GT:GL:GP:GQ:DP:AAC  0/0/0/0/0/0/0/0/0:-14.24,-21.3,-158.59:255,0,0:255:74:0,64,0,6
chrI    103 .   C   G   255 .   CNV GT:GL:GP:GQ:DP:AC   ./././././././././.:-91.88,-70.51,-178.33:0,0,0:0:93:52,17
chrI    105 .   A   T   46  .   CNV GT:GL:GP:GQ:DP:AAC  0/0/0/0/0/0/0/0/0:-18.05,-34.96,-224.75:255,0,0:255:99:89,4,0,5
chrI    107 .   C   A   61  .   CNV GT:GL:GP:GQ:DP:AAC  0/0/0/0/0/0/0/0/0:-5.41,-32.22,-257.63:255,0,0:255:107:2,104,0,0
chrI    114 .   T   A,C 255 .   CNV GT:GL:GP:GQ:DP:AC   0/0/0/0/0/0/0/0/0:-17.92,-40.74,-313.82,-55.96,-315.92,-331.13:255,0,0,0,0,0:255:137:127,7,0
chrI    115 .   C   A,T 255 .   CNV GT:GL:GP:GQ:DP:AC   0/0/0/0/0/0/0/0/0:-64.89,-82.93,-322.06,-80.76,-306.77,-319.58:154,0,0,0,0,0:255:141:111,7,8
chrI    117 .   A   G   255 .   CNV GT:GL:GP:GQ:DP:AAC  0/0/0/0/0/0/1/1/1:-106.97,-43.47,-247.3:0,255,0:255:150:100,0,43,0
```

After performing this step with the three selected files proceed to click on the List Merge screen and select the new VCF files and click on Merge vcf.



Output file for Merge vcf Files



Final Result for Merge VCF:

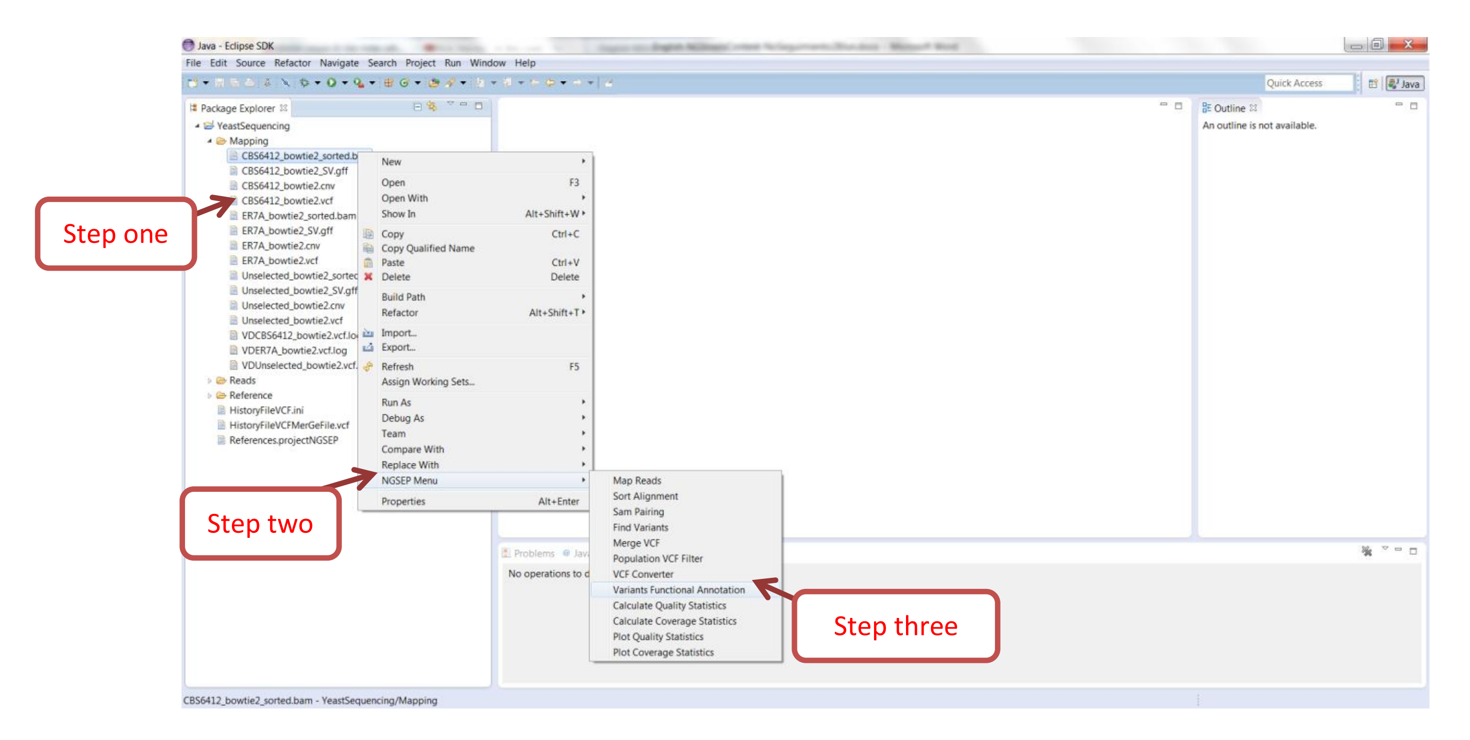
In this final VCF file with genomic variants should see matching each mutation to VCF file for the selected genotype.

Variants Functional Annotator

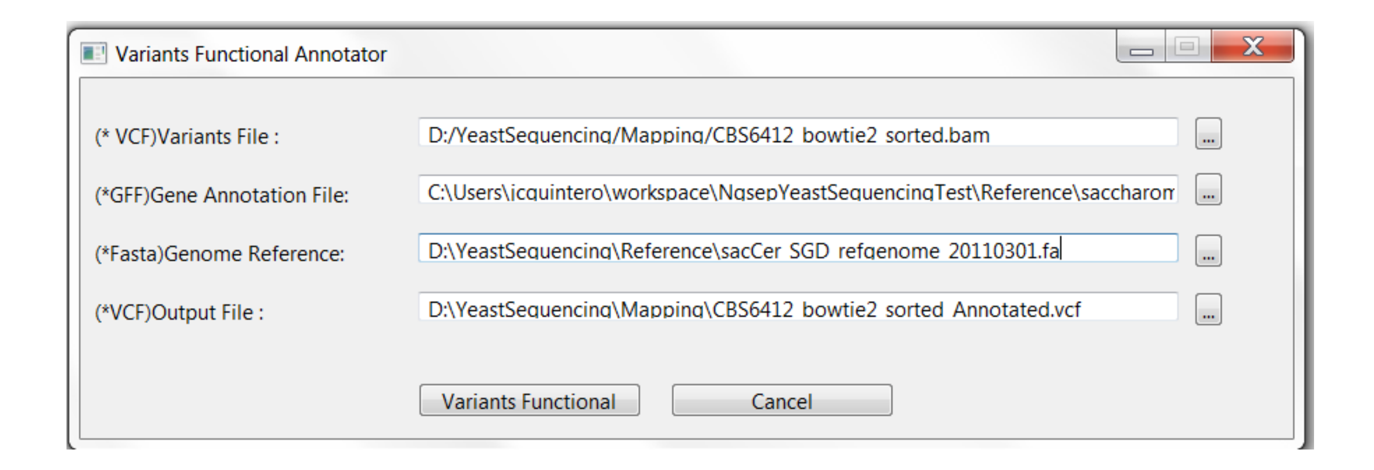
This process is aimed to compare a catalog of genomic variants such as: SNPs and Small indels, with a catalog of gene annotations and a reference genome, will obtain variants but also adding the gene function.

ACCESS TO VARIANTS FUNCTIONAL ANNOTATOR

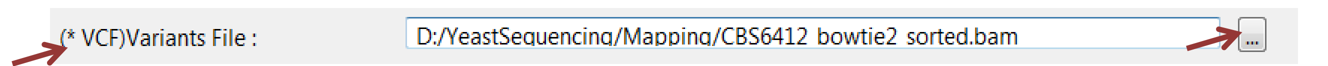
1. The first step in order to access to Variants Functional Annotator after installing Eclipse and NGSEP is having the VCF file (It could be the output of the Variant Detector) .
2. Click on the VCF file, and choose the **Variants functional annotator** option from the NGSEP
3. Make sure that the selected file is a **VCF** File otherwise the process will not work.



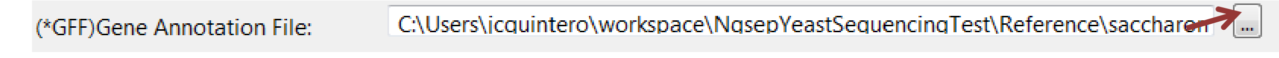
Screen for Variants Functional Annotation



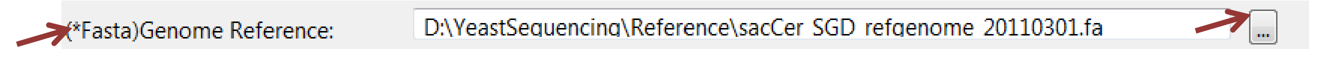
4. **(*VCF) Variants File:** In this field you can see the path of the VCF file that you selected (It could be the output file of the "Variants Detector" function. You can also use the browser on the right in case you want to change the input file.



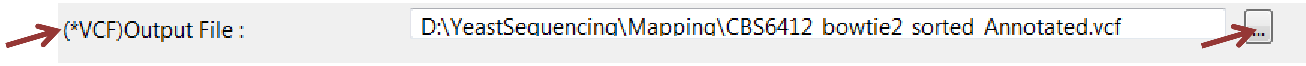
5. **(*)Gene Annotation File:** This field is mandatory because it is the basic input of annotations. At the beginning it will be blank; you have to browse a GFF file with the sample annotations. For further executions the field will display the last file used.



6. **(*Fasta)Genome Reference:** This field is mandatory because the reference genome is going to be used to compare your data. The first time that you execute this functionality this text field will be blank, you must browse for a fasta file with the genome reference. For further executions the field will display the last reference used.



7. **(*VCF)Output File:** In this field you should enter the name and path where you want your output file; we recommend using the same project directory.



8. Use the button with the label Variants Functional Annotator to execute if you want to close the window click on cancel.



Final Result for Variants Functional Annotator:

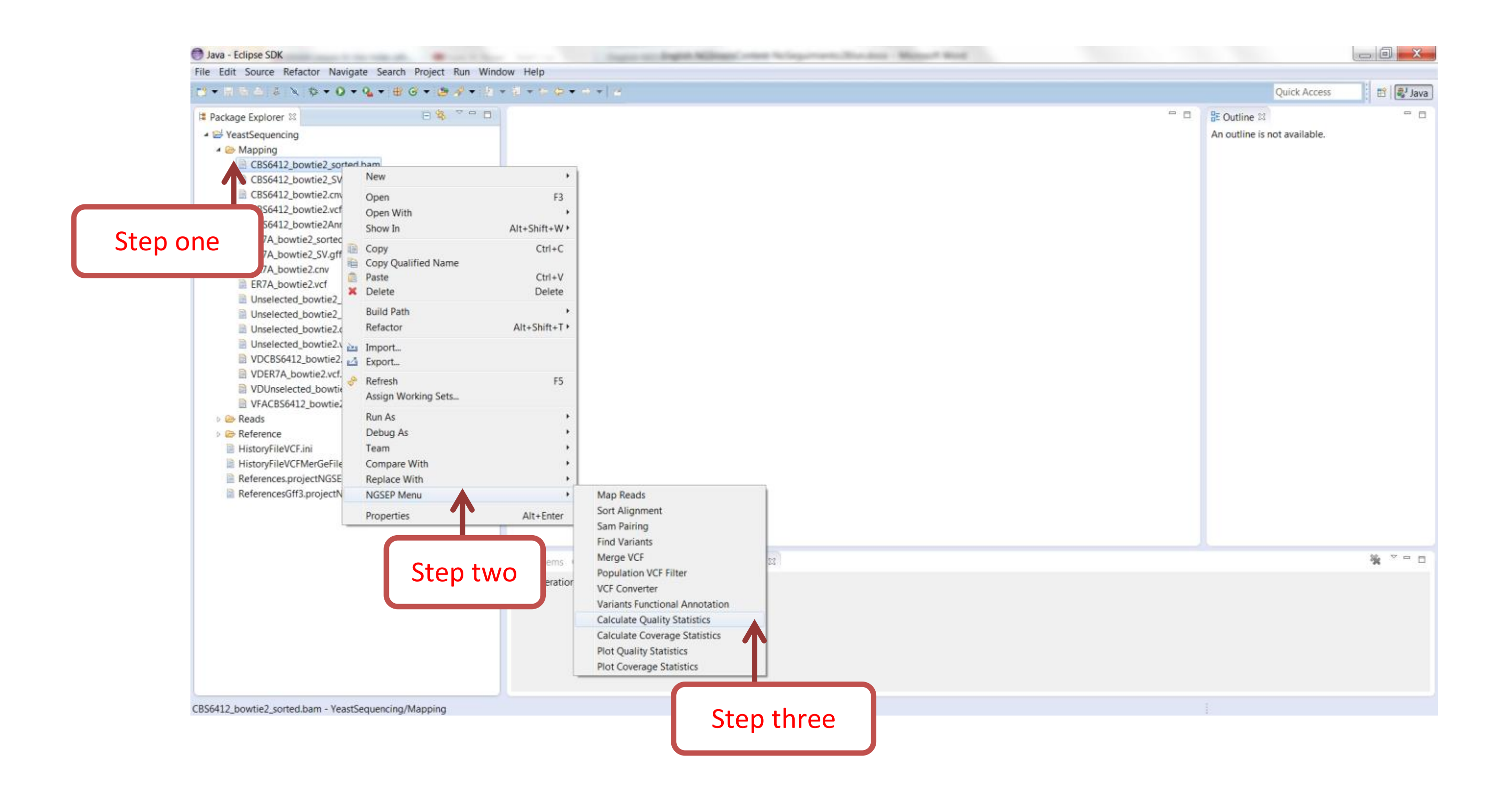
-At the end of this process you will see a VCF file holding the information about genes changes and their variations.

Calculate Quality Statistics

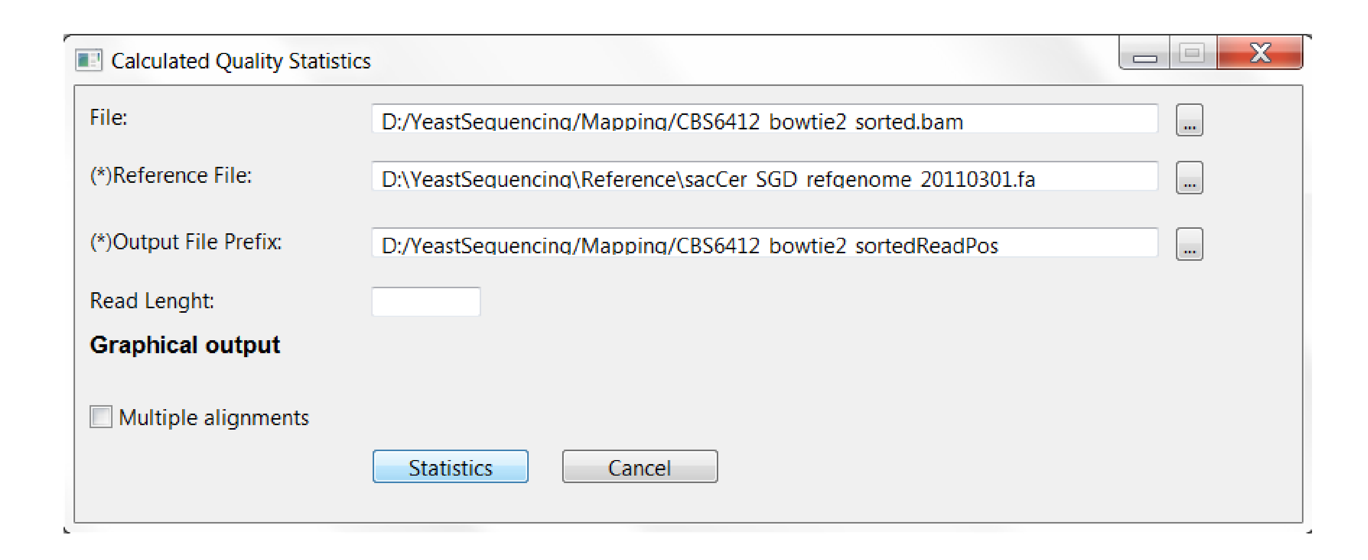
This process basically compares the reads held in the Bam file according to the reference genome, and then NGSEP will be able to indicate the number of sequencing errors for each position of the reads as one set. It should have a homogenous distribution around one.

ACCESS TO CALCULATE QUALITY STATISTICS

1. The first step in order to access to Calculated Quality Statistics after installing Eclipse and NGSEP is having the Sorted Bam file.
2. Click on the Sorted.bam file, and choose the **Calculated Quality Statistics** option from the NGSEP menu
3. Make sure that the selected file is a Sorted Bam File otherwise the process will not work.

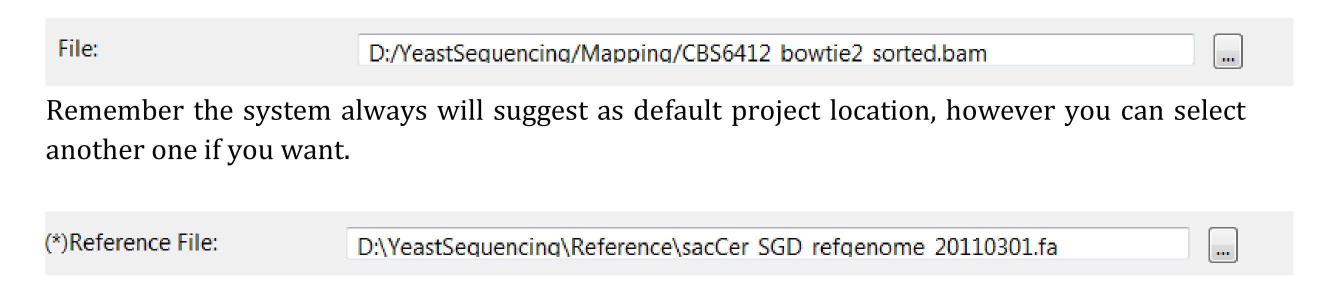


Screen Calculated Quality Statistics



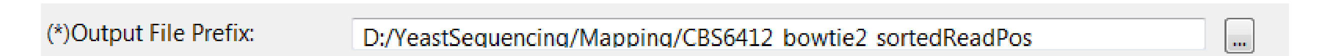
4. This screen is composed by 5 field:

File: In this field you can see the path of the input file that you selected (It could be the output file of the "Sort Alignment" function of NGSEP). You can also use the browser on the right in case you want to change the input file. Our advice is to have all the input files in the project directory.
(*) Reference File: This field is mandatory because the reference genome is going to be used to compare our reads ((*) File). The first time that you execute this functionality this text field will be blank, you must browse for a fasta file with the genome reference. For further executions the field will display the last reference used.

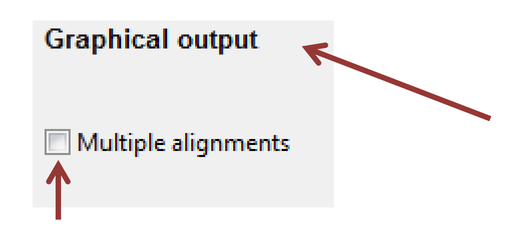


Remember the system always will suggest as default project location, however you can select another one if you want.

(*)Output File: In this field you should enter the name and path where you want your output file; we recommend using the same project directory.



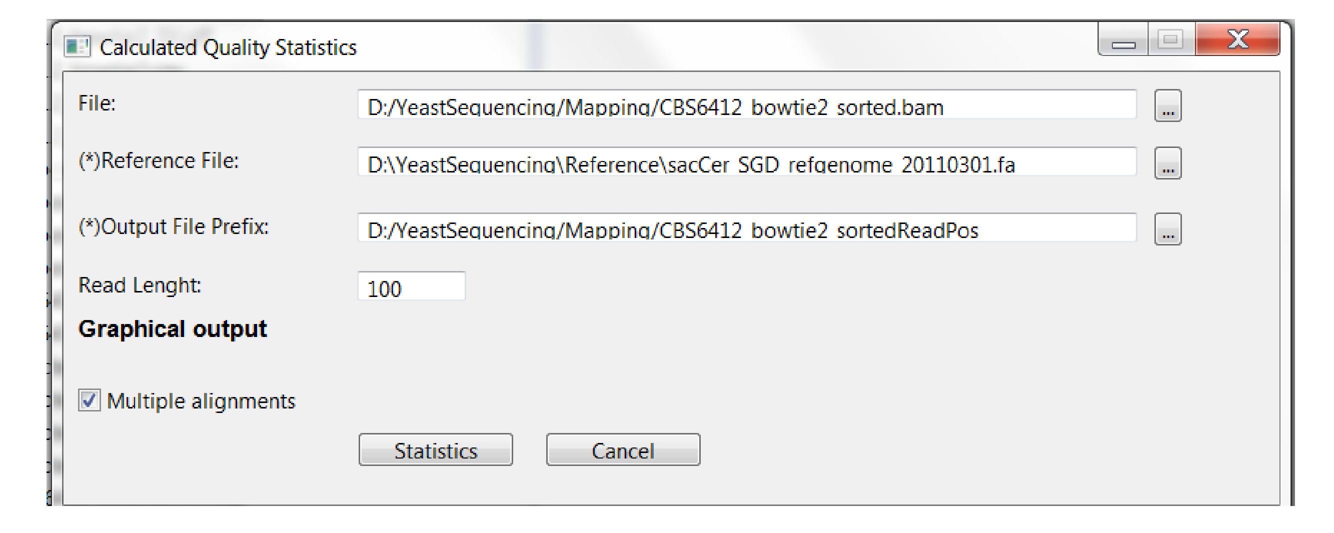
Read Length: Enter the length generated by the sequencer, this number must be an integer. By default the system will consider a length of 100.



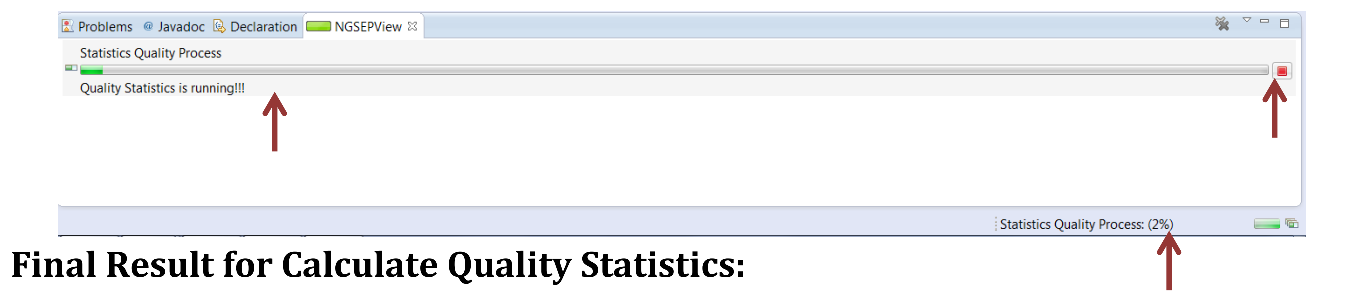
Multiple alignments Choose this option if you want to generate the graphic using multiple alignment data. If you don't choose it, by default the system will take unique alignments.

5. Use the button with the label Statistics to execute if you want to close the window click on cancel.





Note: When you execute the Calculated Quality Statistics, a progress bar will be displayed on the bottom, it represents the percentage of completed process this is important because many times this process can takes several minutes depending on how complex is your organism. If you want to stop the process you are able to do it by pressing the red button in the right side of the progress view. In the end of the process you will see the 2 output files in the directory in the folder that you selected or the default location.

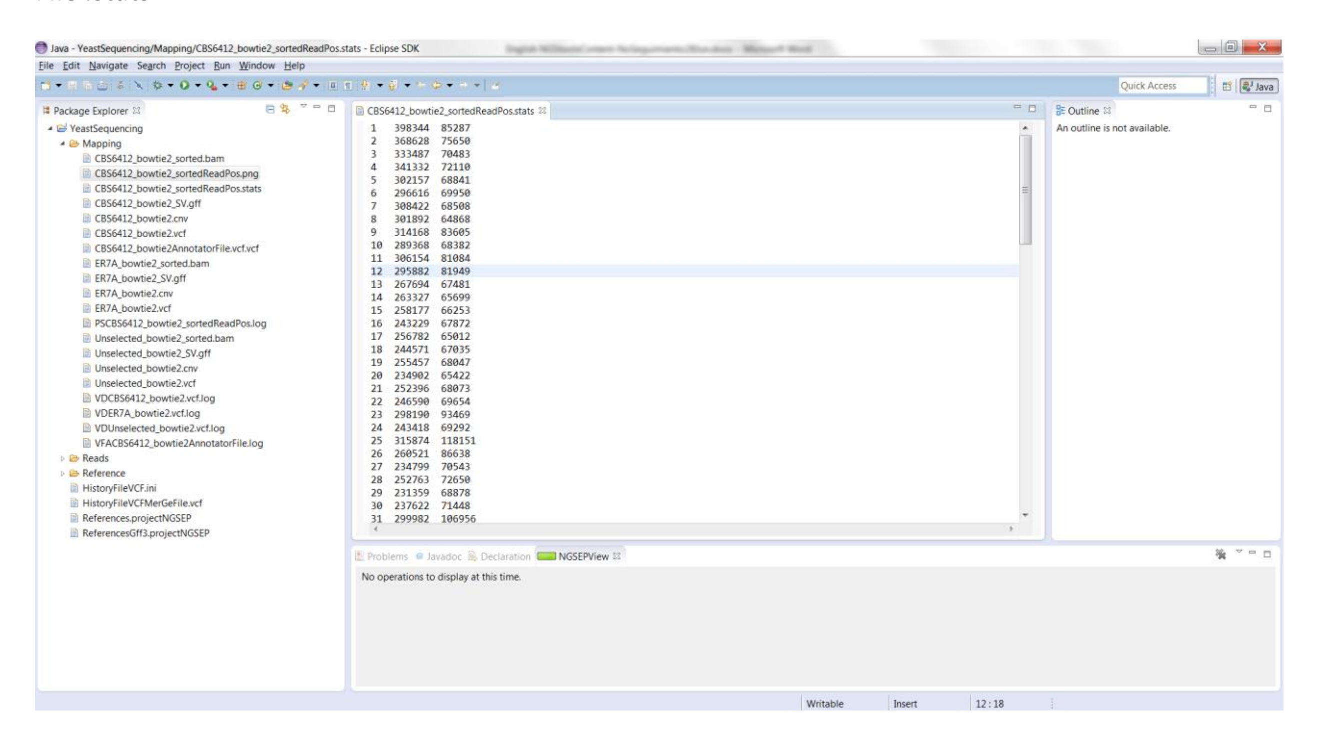


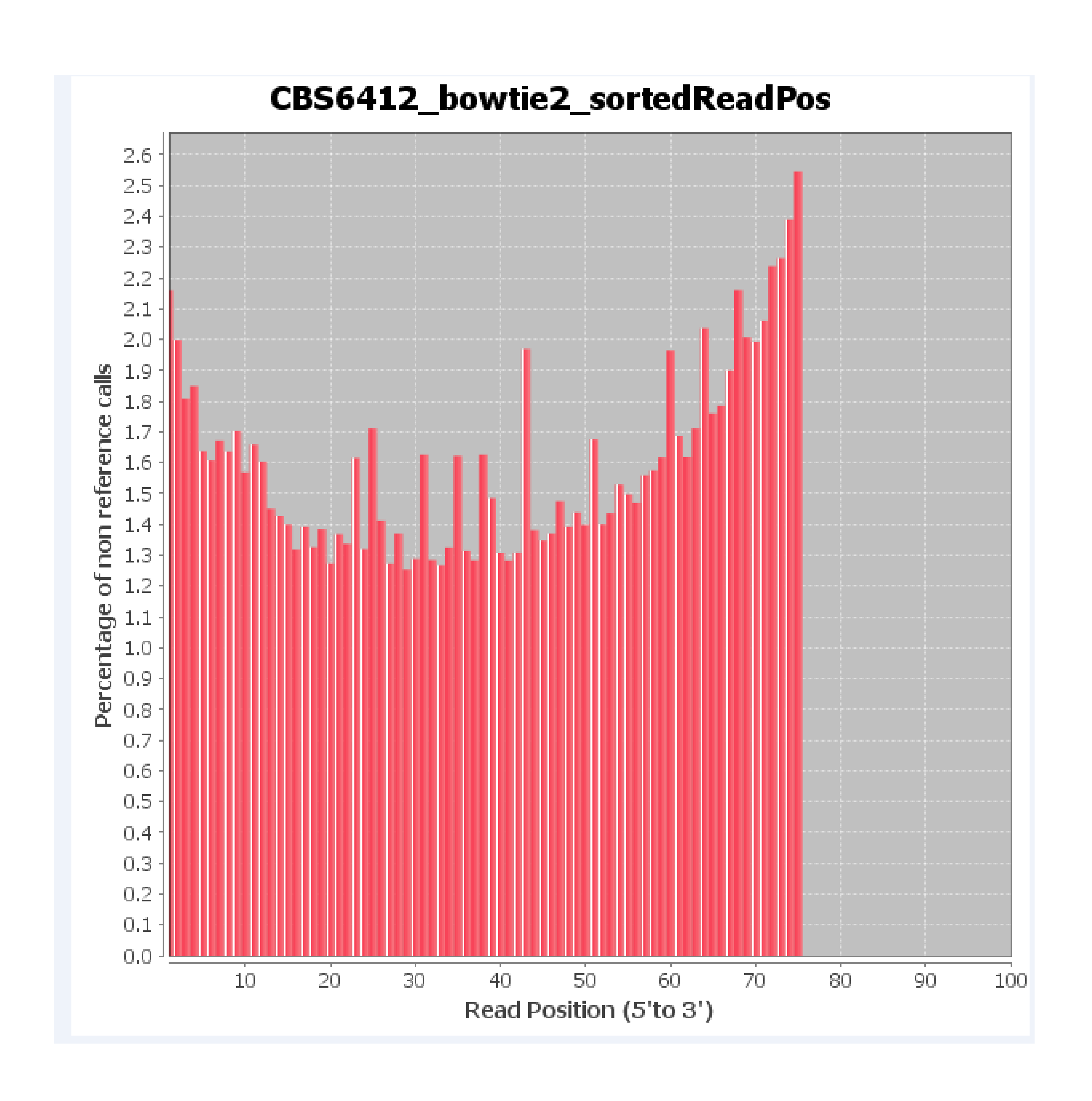
Final Result for Calculate Quality Statistics:

At the end of this process you will generate two files with the same prefix but with different endings. The first file (.stats) holds the statistics of unique and multiple alignments and the second one (.png) is the plot. To open the statistics you can use any text editor and for opening the plot you can use any visual program.

The output quality statistics file will have the format .stats and is tab delimited format composed by 3 columns; first one number of reads, second one number of multiple alignments and third one number of unique alignments. In the end of this file you will find a summary.

File .stats



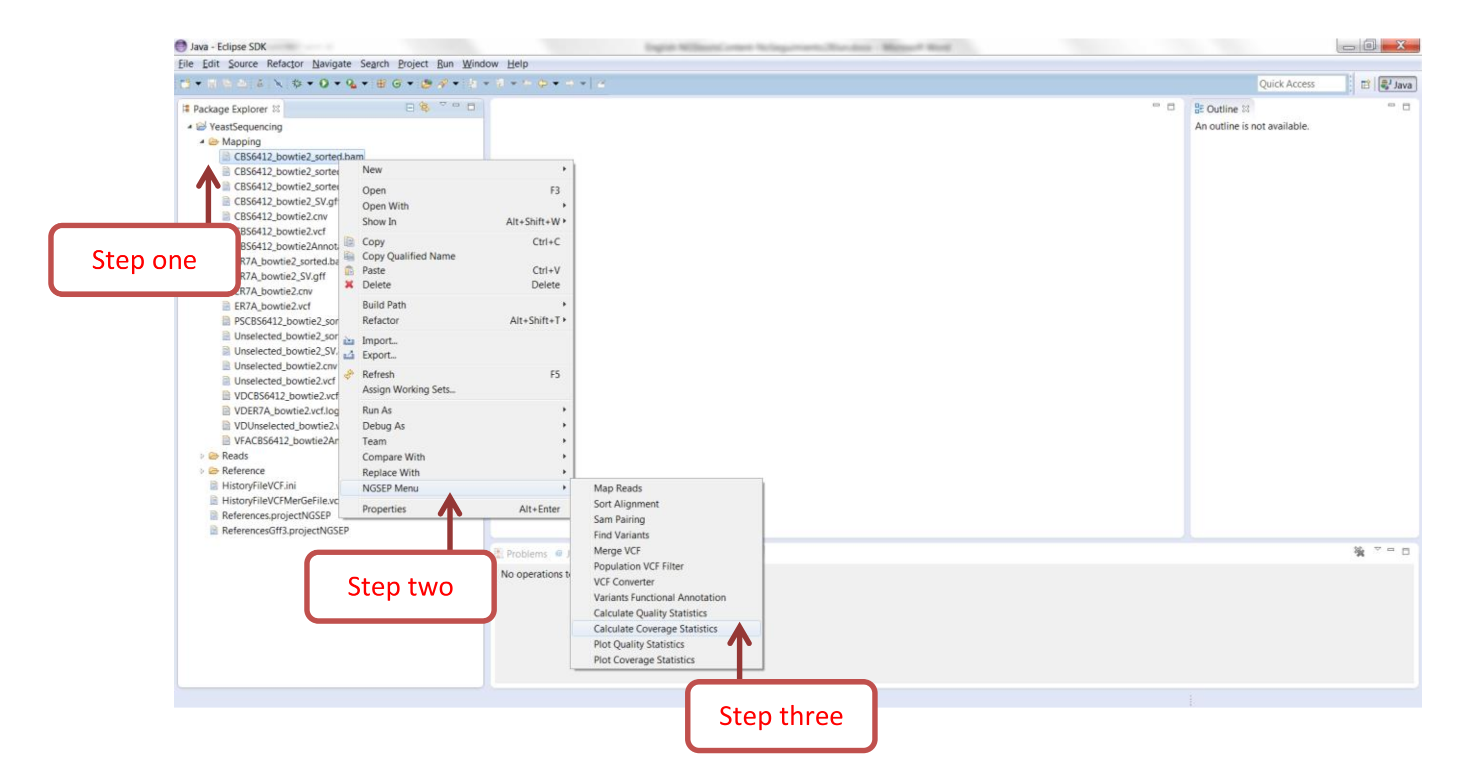

CBS6412_bowtie2_sortedReadPos
Percentage of non reference calls
Read Position (5'to 3')
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
1.1
1.2
1.3
1.4
1.5
1.6
1.7
1.8
1.9
2.0
2.1
2.2
2.3
2.4
2.5
2.6
10
20
30
40
50
60
70
80
90
100

Calculate Coverage Statistics

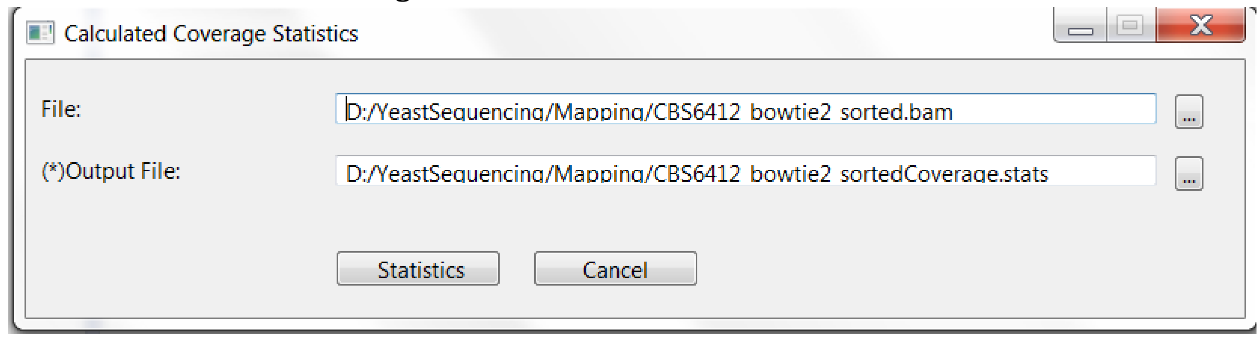
This process compares a reference genome with a sample, looking the number of readings for the sample for covering a position in the reference genome.

ACCESS TO CALCULATE COVERAGE STATISTICS

1. The first step in order to access to Calculated Coverage Statistics after installing Eclipse and NGSEP is having the Sorted Bam file.
2. Click on the Sorted.bam file, and choose the **Calculated Coverage Statistics** option from the NGSEP menu
3. Make sure that the selected file is a Sorted Bam File otherwise the process will not work.



Screen Calculated Coverage Statistics



File: In this field you can see the path of the input file that you selected (It could be the output file of the "Sort Alignment" function of NGSEP). You can also use the browser on the right in case you want to change the input file. Our advice is to have all the input files in the project directory.

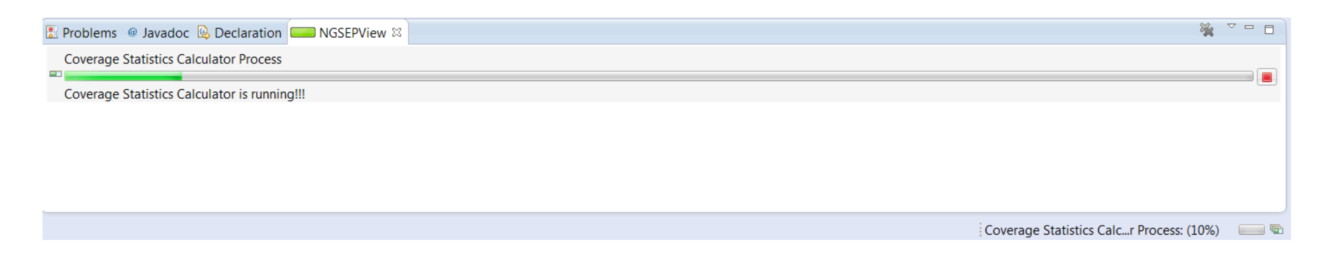
(*)Output File: In this field you should enter the name and path where you want your output file; we recommend using the same project directory.

1. Use the button with the label Statistics to execute if you want to close the window click on cancel.



2.

Note: When you execute the calculated coverage statistics, a progress bar will be displayed on the bottom, it represents the percentage of completed process this is important because many times this process can takes several minutes depending on how complex is your organism. If you want to stop the process you are able to do it by pressing the red button in the right side of the progress view. In the end of the process you will see the 2 output files in the directory that you selected.

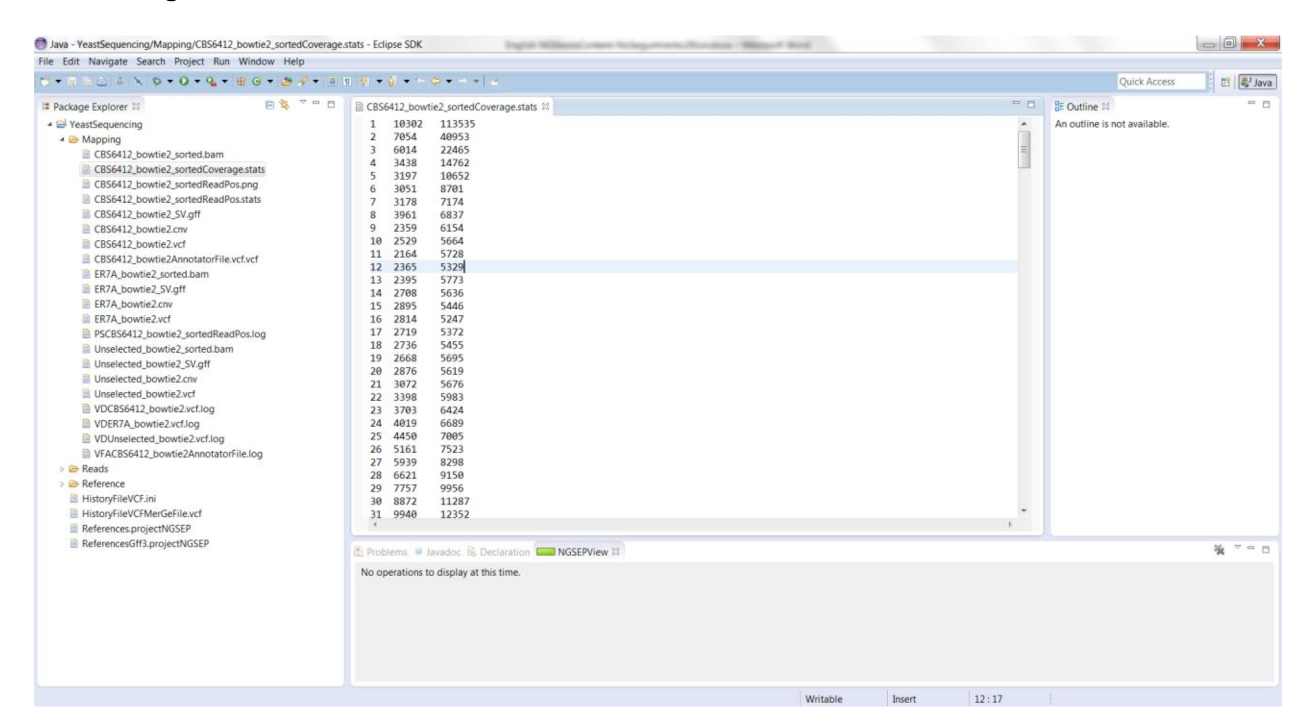


Final Result for Calculate Coverage Statistics:

At the end of this process you will generate one file with the same prefix as your input file but with ending coverage.stats.

The file coverage.stats is a file tab delimited, composed by 3 columns; the first one has the number of reads, the second one the number of multiple alignments and the third one number of unique alignments. In the end of this file you will find a summary.

File coverage.stats



Plot Quality Statistics

With this function you are going to generate a plot from the quality statistics file previously generated in “Calculate Quality Statistics”.

INPUT FILE

- File .stats: Format tab delimited compose by 3 columns, first one number of reads, second one number of multiple alignment and third one number of unique alignments. In the end of this file you will find a summary.

ACCESS TO PLOT QUALITY STATISTICS

1. The first step in order to access to Plot Quality Statistics is having the statistics file generated by the “Calculate Quality Statistics”option.
2. Click on the .stats file, and choose the **Calculated Quality Statistics** option from the NGSEP menu

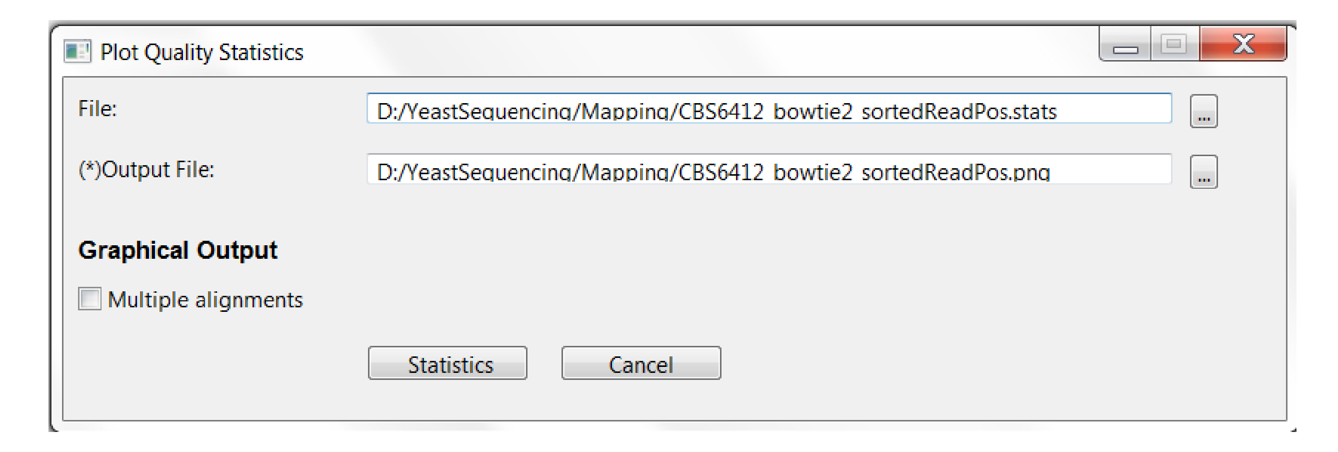
3. Make sure that the selected file is a statistics file otherwise the process will not work.

Step one

Step two

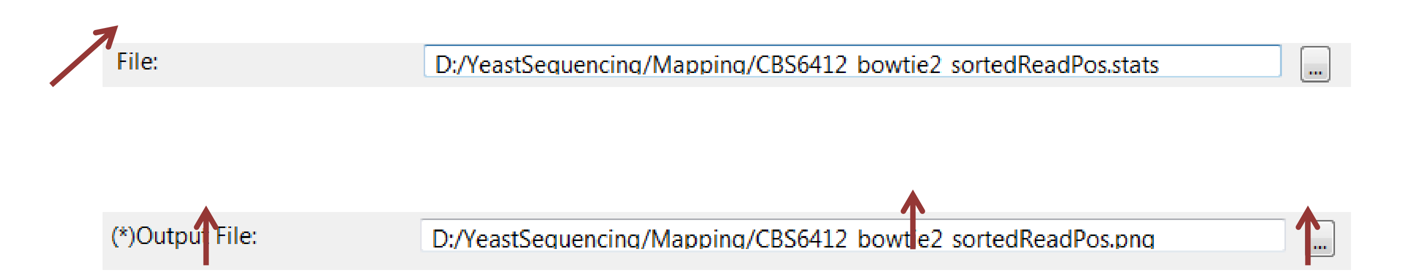
Step three

Screen Plot Quality Statistics

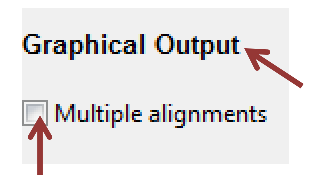


File: In this field you can see the path of the input file that you selected (The output file of the "Calculate Quality Statistics" function of NGSEP). You can also use the browser on the right in case you want to change the input file. Our advice is to have all the input files in the project directory.

(*)Output File: In this field you should enter the name and path where you want your output file; we recommend using the same project directory.

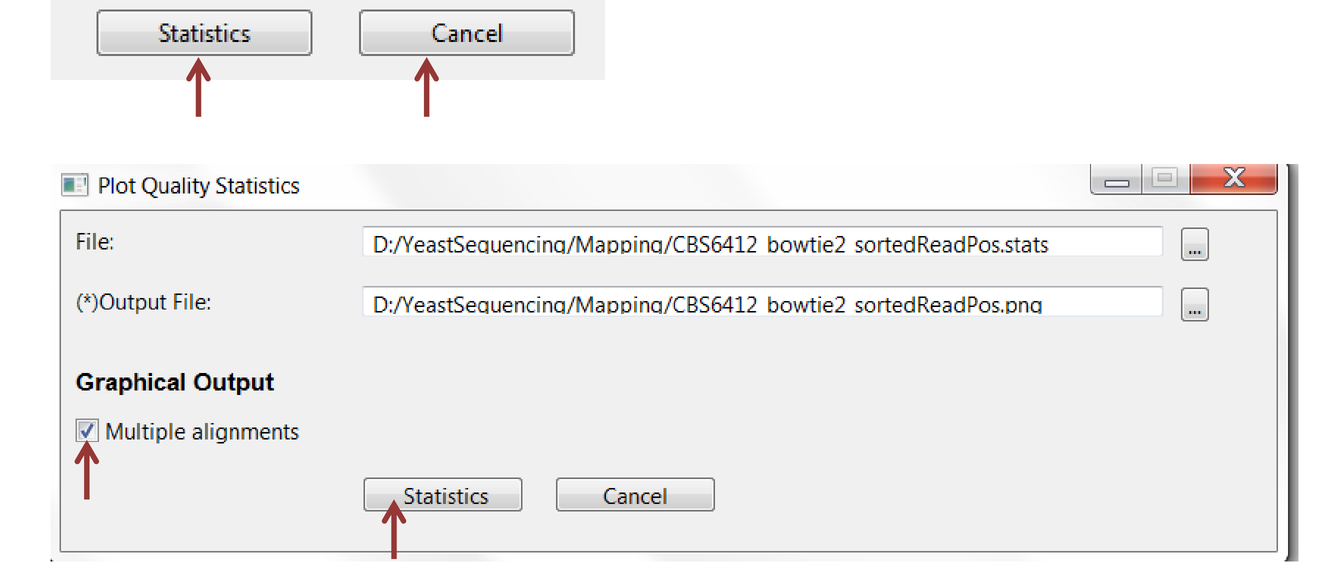


Multiple alignments Choose this option if you want to generate the graphic using multiple alignment data. If you don't choose it, by default the system will take unique alignments.



4. Use the button with the Plot Quality Statistics to execute if you want to close the window click on cancel

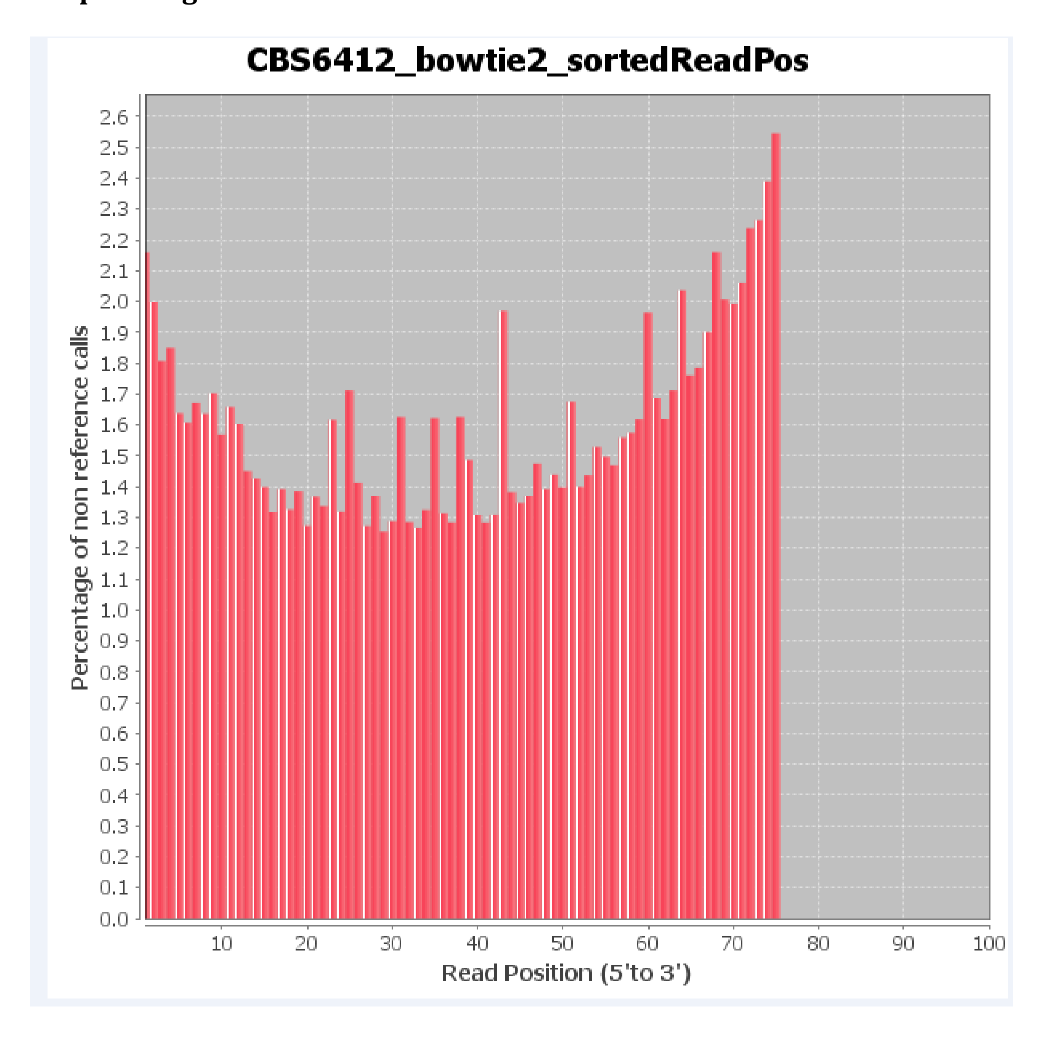
Statistics Cancel



Final Result for Plot Quality Statistics:

At the end of this process you will generate a file .png. To open it you can use any visual program. The x axis represent the Read Position (From 5'to 3'), and the Y axis the Percentage of non-reference calls.

Output image



Plot Coverage Statistics

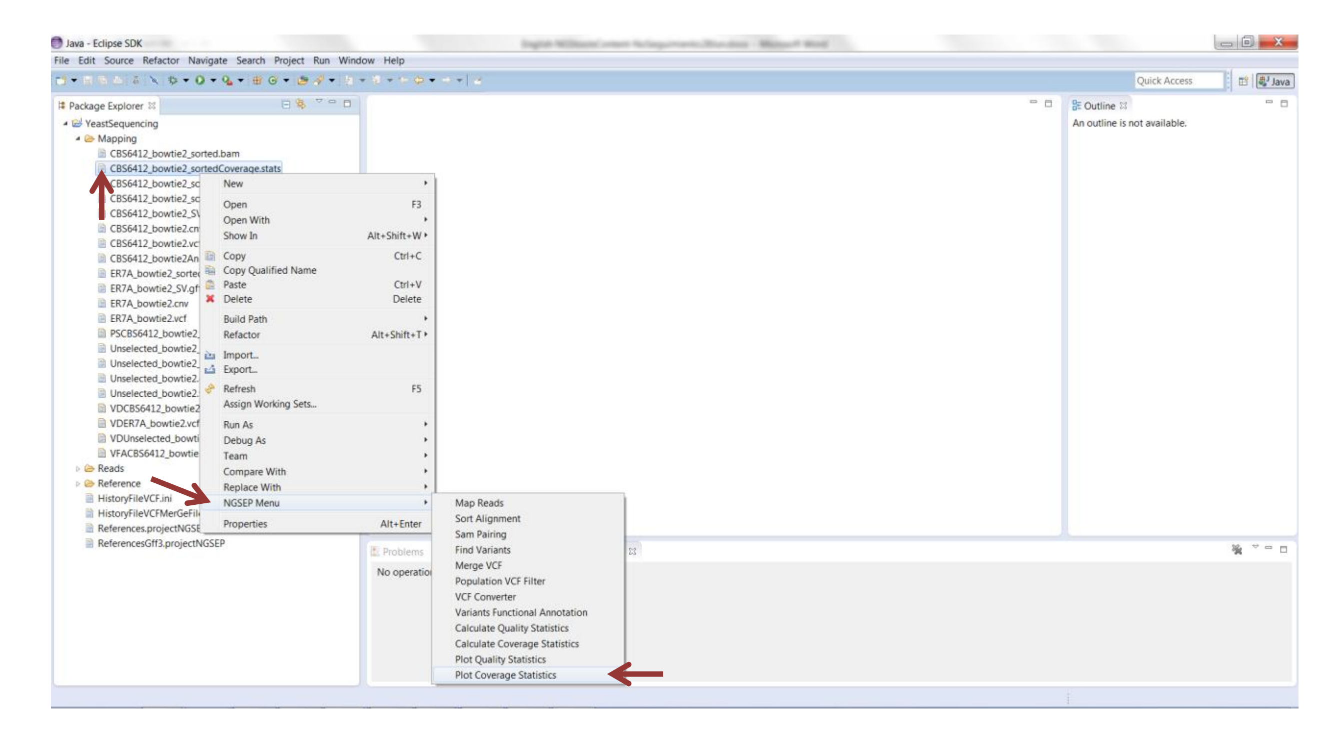
With this function you are going to generate a plot based on the file Coverage.stats that holds the data about the coverage for each position, considering unique and multiple alignments. It should have a normal distribution centered on the expected Coverage.

INPUT FILES

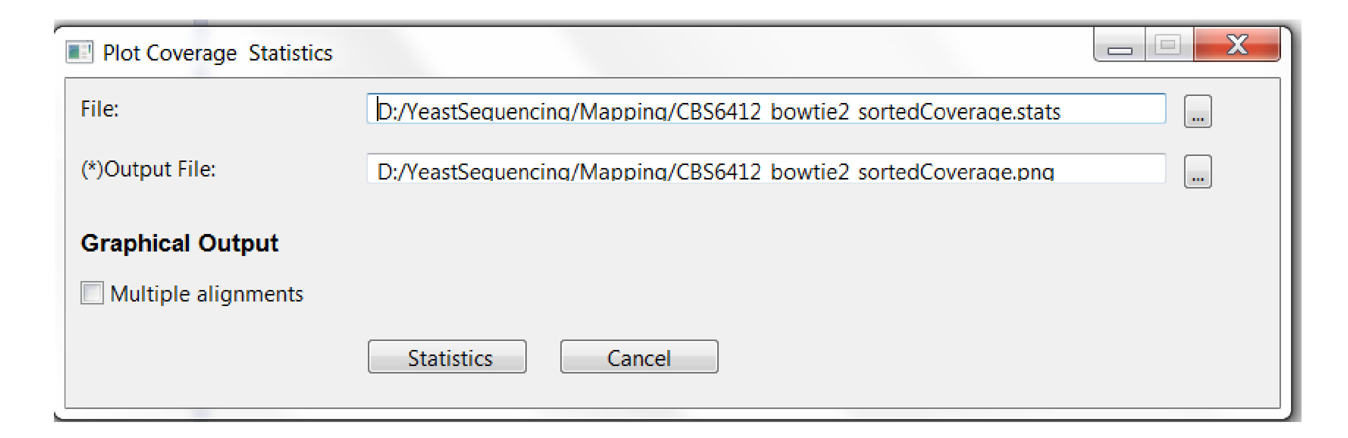
File coverage.stats: Is tab delimited file, composed by 3 columns; the first one has the number of reads, the second one the number of multiple alignments and the third one number of unique alignments. In the end of this file you will find a summary.

ACCESS TO PLOT COVERAGE STATISTICS

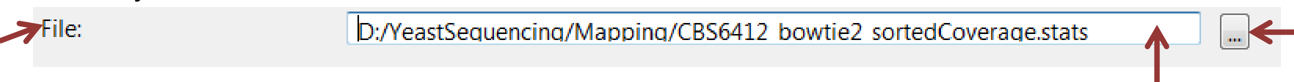
1. The first step in order to access to Plot Coverage Statistics is having the coverage.stats file generated by the "Calculate Coverage Statistics" option.
2. Click on the coverage.stats file, and choose the **Plot Coverage Statistics** option from the NGSEP menu.
3. Make sure that the selected file is a statistics file otherwise the process will not work.



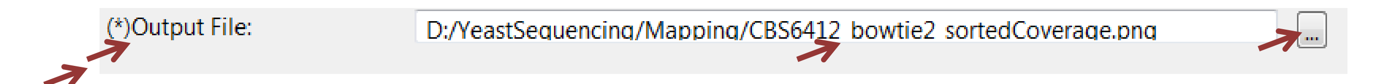
Screen Plot Coverage Statistics



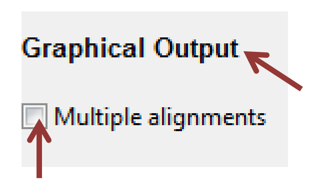
File: In this field you can see the path of the input file that you selected (The output file of the "Calculate Quality Statistics" function of NGSEP). You can also use the browser on the right in case you want to change the input file. Our advice is to have all the input files in the project directory.



(*)Output File: In this field you should enter the name and path where you want your output file; we recommend using the same project directory.

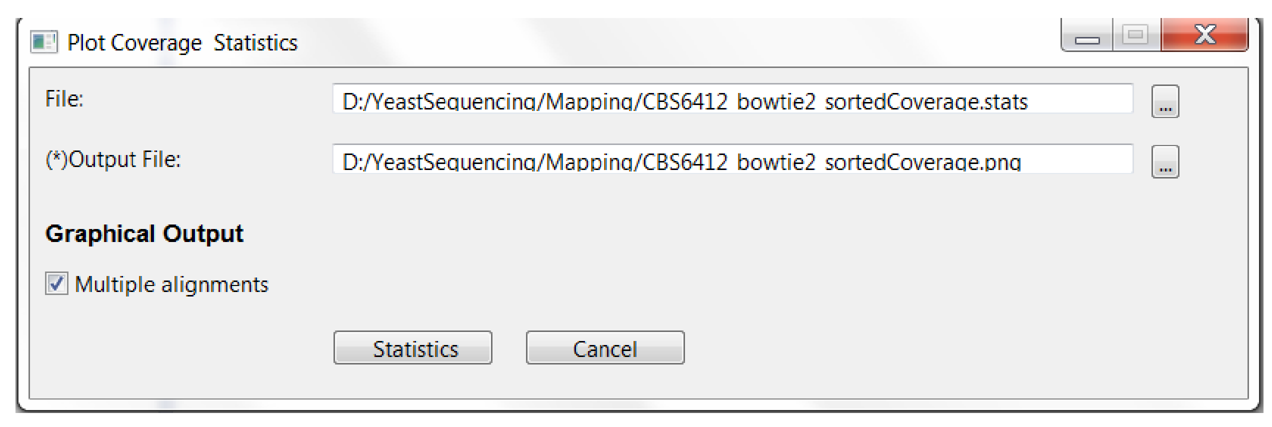


Multiple alignments Choose this option if you want to generate the graphic using multiple alignment data. If you don't choose it, by default the system will take unique alignments.



5. Use the button with the label Statistics to execute if you want to close the window click on cancel.

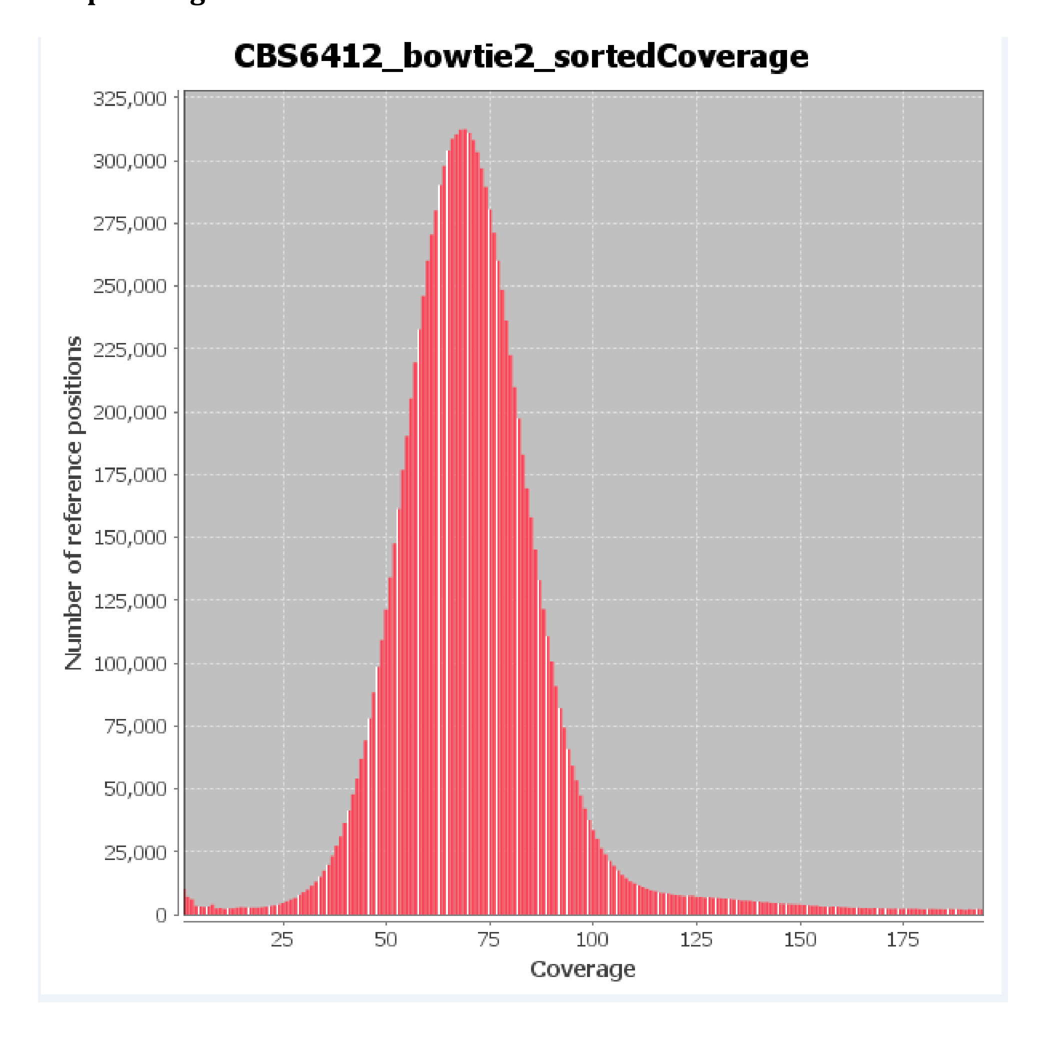




Final Result for Plot Coverage Statistics:

At the end of this process you will generate a file .png . To open it you can use any visual program. The x axis represent the coverage and the Y axis the number of reference positions.

Output image:



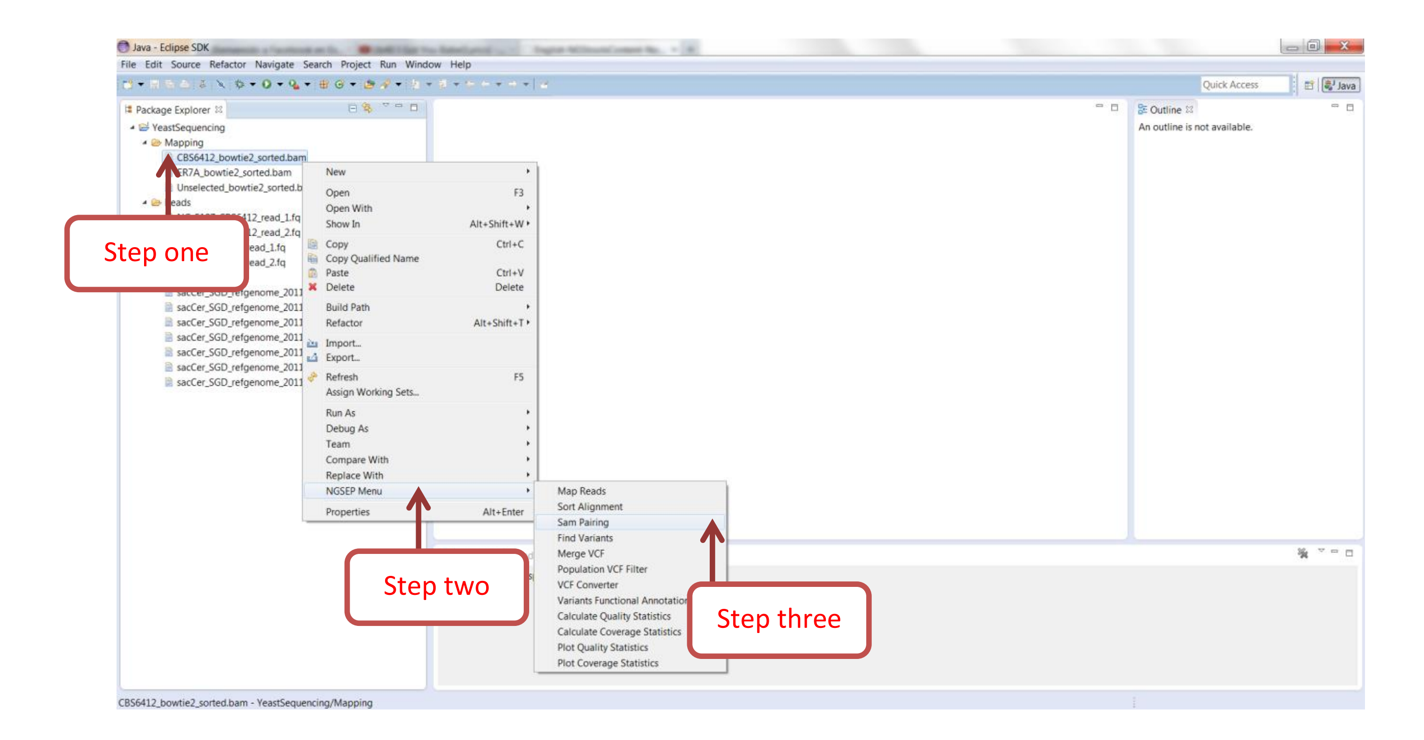
Optional Process

Sam Pairing

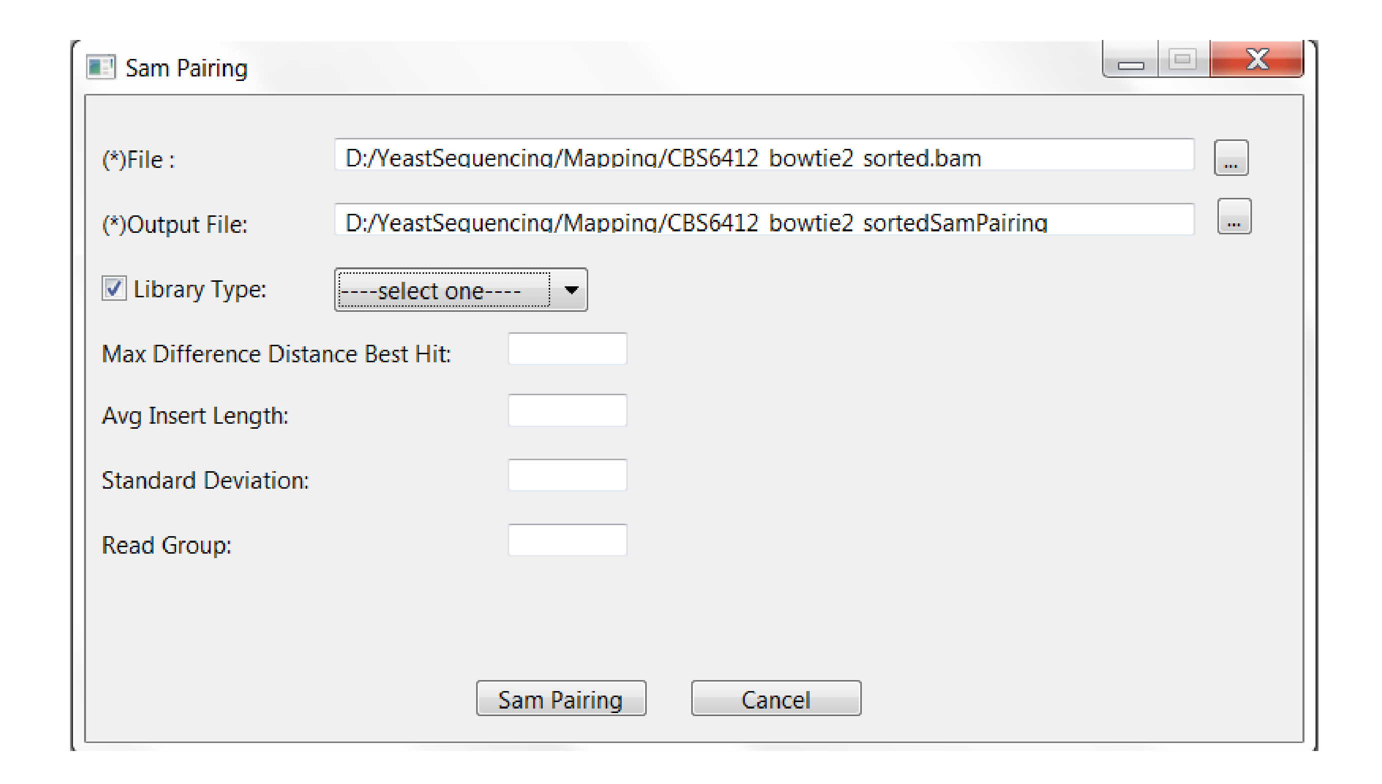
With this function, you will define the couples of reads that match in the same section of the genome according to an insert length defined.

ACCESS TO SAM PAIRING

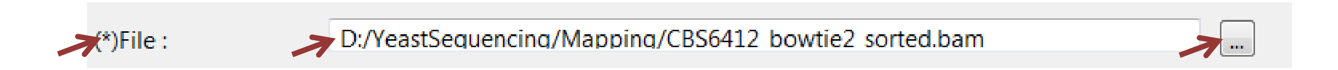
1. The first step in order to access to Sam Pairing after installing Eclipse and NGSEP is having the Sorted Bam file.
2. Click on the sorted .Bam file, and choose the **Sam Pairing** option from the NGSEP menu.
3. Make sure that the selected file is a Sorted Bam File otherwise the process will not work.



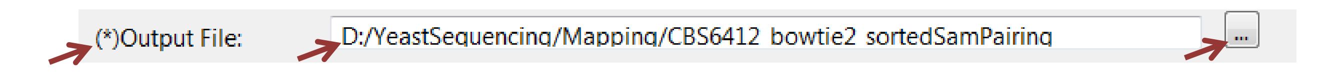
Screen Sam Pairing



4. The first field **(*)File**, holds a text field with the path of the selected file. However you can also use the browser on the right in case you want to change the input file.

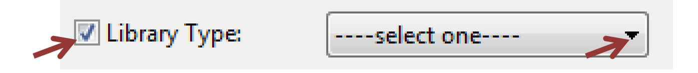


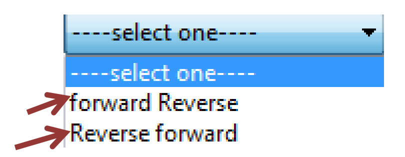
5. Below you will find **(*)Output File:** this text field holds the output file that you browsed, we recommend using the same project directory.



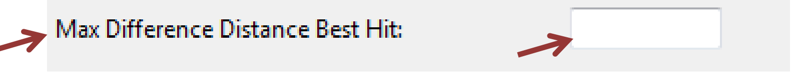
6. **Library Type:** When you choose this option, you have to select from: Forward reverse and Reverse Forward.

   - **Forward reverse:** used when the insert length is less than 1000 is called **paired end**.
   - **Reverse forward:** used when the insert length is more than 1000 is called **mate pair**.

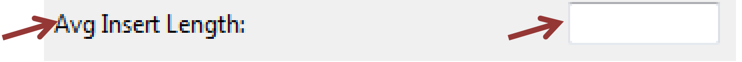




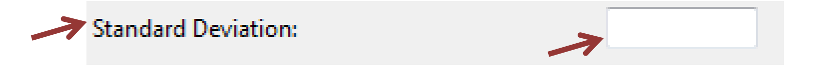
7. **Max Difference Distance Best Hit:** In this field enter the number of the maximum distance accepted among the best hit and the rest. The best is taking into account as the position of a couple of reads in a genome with netter acceptance.



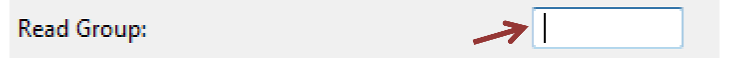
8. **Avg Insert Length:** Enter the size of the fragments that you hired. The default is 500 bps.



9. **Standard Deviation:** Enter a number that represents a measure of dispersion, which means how much can the values move away from the average entered in the previous field (**Avg Insert Length**).



10. **Read Group:** Enter the name of the set of reads for the output files.



11. Use the button with the label Sam Pairing to execute if you want to close the window click on cancel.



Final Result for Sam Pairing:

-At the end of this process you will see a Bam file with the best sets of paired reads.